11-2008

Optimal Designs of Mobile Nuclear Engines to Power Manned Vehicles On Mars

Robert D. Woolley
University of Tennessee - Knoxville, woolley@pppl.gov

To the Graduate Council:

I am submitting herewith a thesis written by Robert D. Woolley entitled "Optimal Designs of Mobile Nuclear Engines to Power Manned Vehicles On Mars." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Nuclear Engineering.

Fred R. Mynatt, Major Professor

We have read this thesis and recommend its acceptance:

Laurence F. Miller, Ronald E. Pevey, Arthur E. Ruggles

Accepted for the Council:
Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

To the Graduate Council:

I am submitting herewith a thesis written by Robert D Woolley entitled "Optimal Designs of Mobile Nuclear Engines to Power Manned Vehicles On Mars." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Nuclear Engineering.

Fred R. Mynatt, Major Professor

We have read this thesis
and recommend its acceptance:

Laurence F. Miller

Ronald E. Pevey

Arthur E. Ruggles

Accepted for the Council:

Carolyn R. Hodges
Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Optimal Design of Mobile Nuclear Reactor Engines to Power Manned Vehicles On Mars

A Thesis Presented for
the Nuclear Engineering
Master of Science Degree

The University of Tennessee, Knoxville

Robert D. Woolley
November 2008

DEDICATION

This work is dedicated to the future generations of human visitors to Mars who will harness nuclear power to explore the martian surface and exploit the planet's resources, in the service of science and of life.

ACKNOWLEDGEMENTS

I thank my thesis advisor Dr. F.R. Mynatt and the members of my faculty review committee, Dr. L.F. Miller, Dr. R.E. Pevey, and Dr. A.E. Ruggles, for their detailed professional review and guidance.

I thank my wife Kathryn for her patience during this time-consuming project and for her editorial help.

ABSTRACT

This work develops original conceptual designs for compact nuclear fission reactor engines to power robust mobile equipment operating on the surface of the planet Mars. This is a nuclear application area not well explored in previous publications. Some novel analytical approaches are developed herein, including the application of optimal control theory to minimize radiation shielding mass. This work also provides the first study of using another planet's atmosphere to implement open-cycle thermal conversion systems.

To power equipment on Mars for extended durations at sustained power levels ranging from one hundred horsepower to several thousand horsepower, there is no practical alternative to a nuclear fission heat source. Design difficulties arise from mobility's need to restrict engine size and mass, each of which is, in turn, determined by the schemes chosen for thermal conversion waste heat rejection and for neutron and gamma radiation shielding.

The conceptual design solutions pursued herein entirely avoid a large waste heat rejection radiator or low pressure heat exchanger by instead using the martian air directly as the thermal conversion fluid. This Open Brayton Cycle implementation unconventionally employs large-diameter radial-flow compressor/turbine designs for the lower pressure air-flow stages in order to obtain sufficient efficiency from the low pressure martian air. Design prescriptions and analyses for these rotating components are included.

The radiation shielding mass has been minimized by numerical algorithms developed as part of this work to solve the Euler-Lagrange equations for a minimum mass shield meeting stated radiation leakage requirements. In addition, a risk-balancing approach is taken to setting those radiation requirements in order to avoid excessive conservatism.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
1 INTRODUCTION AND OVERVIEW

The overall systems design chosen for nuclear engines to use on the martian surface configures each engine as an independent, self-propelled modular vehicle. Each engine can be connected through a power-transmitting coupling to different vehicles or other mobile surface equipment to form a tandem train of mobile vehicles. Tandem stability is provided through servo feedback.

Equipment to be powered by these engines includes the following:
(1) large pressurized surface vehicles for manned excursions of extended duration,
(2) ground excavation vehicles (e.g., bulldozers) to construct pressure-tight underground rooms for either rocket fuel storage or human occupancy,
(3) assorted power shovels such as backhoes and trenchers suitable for near-surface mining of water ice from the martian soil,
(4) portable drilling rigs to extract soil samples from deep underground, and
(5) electrical chargers to remotely renew batteries for portable power tools, for tele-operated robots, for unpressurized all-terrain vehicles, and for similar devices.

Figures 1-1 and 1-2 depict one of the conceptual nuclear engine designs resulting from this work, this one developing a rated maximum 1,000 horsepower of mechanical output power. Each engine is mobile with its own computer, sensors, video cameras and radio communications. Each engine can be tele-operated from Earth or from a manned Mars base. Each engine will be delivered by rocket to the martian surface as a separate payload. Some engines will be available on Mars before manned missions begin. Figure 1-1 depicts in plan view the mobile nuclear engine powering a manned surface vehicle, and Figure 1-2 depicts it in side view, showing also a nearby suited astronaut in order to illustrate the size scale.

For mobile nuclear engines powering manned surface vehicles and other equipment on Mars to be practical, a two-pronged engineering design approach must be pursued to restrict their mass. One prong is to minimize the nuclear reactor's physical size and optimize its location in relation to people in order that the massive radiation shielding surrounding it can be minimized. The other prong is to develop an air-breathing open-cycle thermal conversion system which efficiently uses the martian air as its internal thermodynamic working fluid in order to avoid the massive external heat rejection equipment required for closed-cycle systems.

A further complication of this design project is the fact that its focus is an application on an alien world. In the design process I have attempted to assess the state of knowledge about the physical conditions which will be encountered on Mars and to identify many of the essential tasks involved in early manned exploration of that planet. To the extent possible, I have also considered the practical problems which will be encountered in

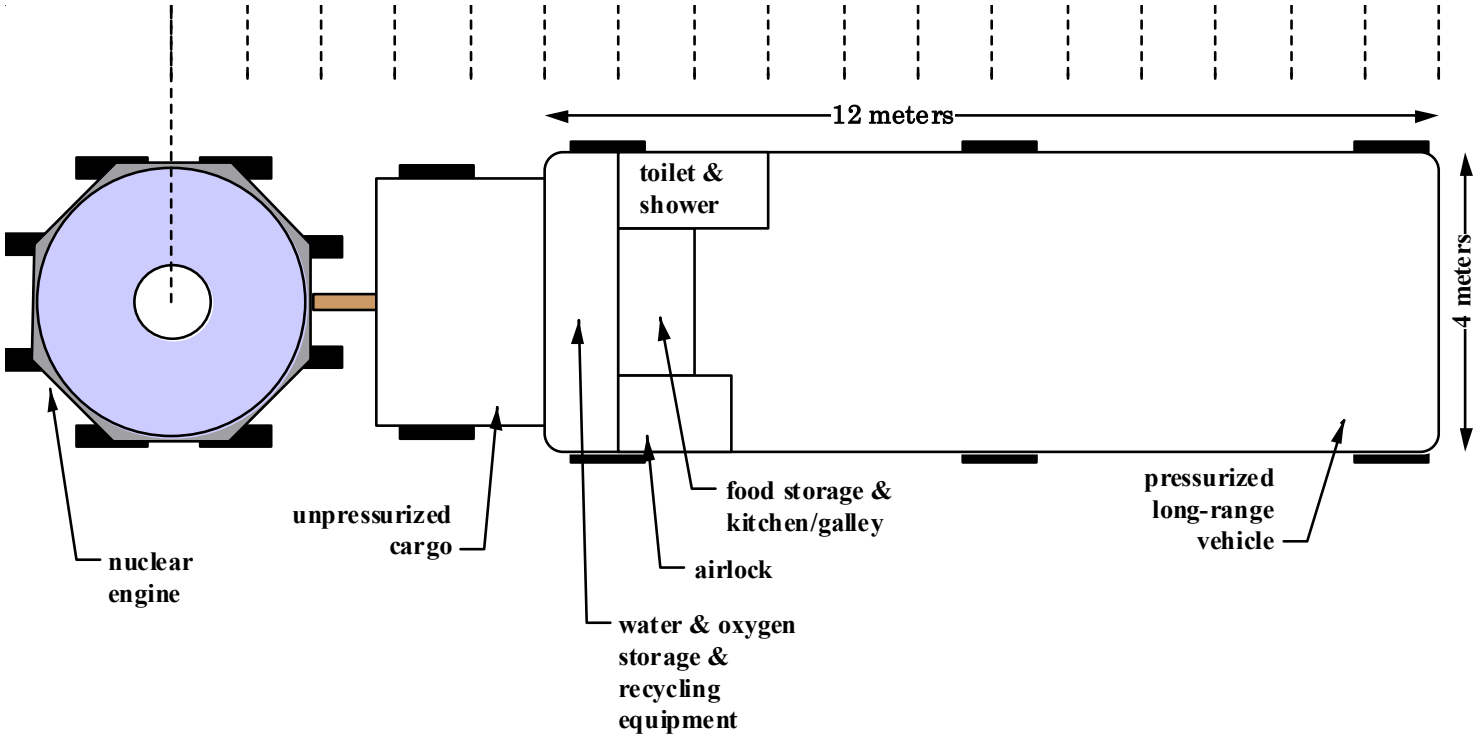


Figure 1-1: Mobile Manned Surface Vehicle With 1,000 Hp Nuclear Engine- Plan View

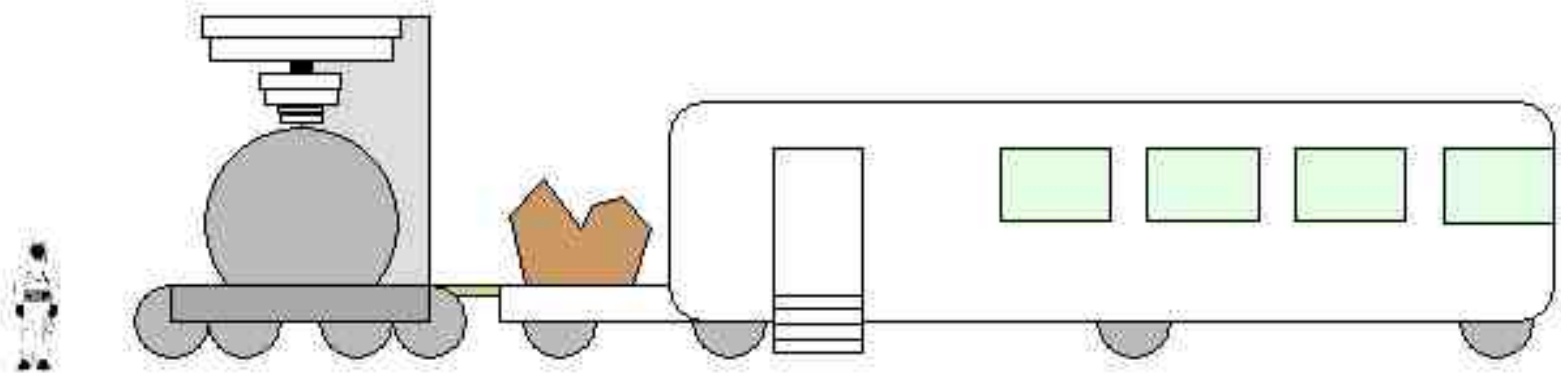


Figure 1-2: Mobile Manned Surface Vehicle With 1,000 Hp Nuclear Engine- Side View

transporting to Mars the unavoidably massive, yet to my mind mission-critical, mobile nuclear engine power plants. Therefore, additional chapters are included containing this resulting background material, i.e., details of the martian environment, the scientific mission rationale, and transportation constraints. These establish the context and boundary conditions from which project requirements have been drawn, and are therefore an essential adjunct to the engineering development of the design. Chapter 2 presents technical facts about the martian environment selected based on their relevance to the design or use of nuclear powered mobile engines by astronauts on Mars. Characteristics of the martian air are particularly important for the design of air-breathing engines. Martian topography defines engine mobility needs. Background radiation levels and the scattering characteristics of surface minerals influence the design of radiation shielding. Martian water is also detailed since it will underlie many activities using the nuclear powered engines on Mars' surface for scientific research, for life support, and for producing rocket fuel needed to return to Earth. Chapter 3 and Appendix A together present the mission context for these nuclear engines including their transportation to Mars, the needs they will address there, who will use them, when, how, in what circumstances, and why.

Nuclear systems engineering aspects of this project are distributed between chapters 4 through 11. Chapter 4 presents overall configuration choices with a detailed focus on the nuclear configuration, while chapter 5 presents preliminary reactor criticality studies using discrete ordinates codes from the SCALE system. Chapter 6 considers nuclear radiation protection criteria appropriate for manned mars missions and translates them into radiation shielding requirements for the nuclear powered engines. Chapter 7 presents initial radiation shielding calculations using discrete ordinates codes from the SCALE system. Chapter 8, which is a central focus of this thesis, presents the original derivation as part of this work of Euler-Lagrange equations for the minimum-mass design of radiation shielding using optimal control theory, the development of numerical algorithms to solve them, and the implementation of those algorithms in a new SCALE software module, also coded as part of this work. Chapter 9 presents results of studies using this new SCALE module to consider families of different optimal minimum-mass radiation shield designs. Chapter 10 briefly discusses shielding studies modeling the full 3D configuration using Monte Carlo code systems. These have not been done but should be done next. Chapter 11 briefly discusses the analysis of sensitivities connecting design parameters of performance and mass that also should be done to support tradeoff studies between design options.

Chapters 12 through 15 contain engineering analyses relevant to design of the engine's Open Brayton Cycle thermal conversion system. Chapter 12 generally reviews thermal conversion system fundamentals, makes the point that waste heat rejection is a severe design bottleneck for mobile engines, and shows that the martian air can conveniently accept that heat without any low pressure heat exchanger as the exhaust from open cycle thermal conversion systems. It also analyzes the alternative of relying on a true thermal radiator, showing that approach would unduly limit engine power levels.

Chapter 13 presents analyses of the Open Brayton Cycle using the actual nonlinear thermodynamic properties of martian air and develops relationships between the temperature range, compression ratio, efficiency and specific power of alternative engine designs using ideal zero-loss compressor, turbine, and heat transfer components. Using polytropic component models, it also derives limits on the allowable inefficiencies of those engine components.

Chapter 14 reviews turbine/compressor theory and practice along with the theory of dynamic similitude. It presents the turbocompressor component selection process which relies largely on dimensional similitude methods but which results in detailed designs. Using published correlations from references describing actual non-ideal turbine/compressor performance, the turbomachine design parameters for adequate component efficiency during operation on Mars are chosen.

Chapter 15 presents design analyses for the high pressure liquid-to-gas heat exchanger using extended surfaces (i.e., fins) which will transfer heat from the reactor coolant into the compressed martian air.

In addition to the Appendix A discussions of relevant technological parameters forming boundary conditions for this project, multiple other appendices are provided to document important calculations which are also too long to include in the body of this thesis. Appendices B, C, D, E, and F are configured in the combined MSWORD/MATLAB Notebook form which is convenient for self-documenting computations.

It is appropriate to mention here that there has been a continuing substantial public interest in prospects for manned missions to Mars, and that interest has led to a panoply of relevant technical studies carried out by individuals, by voluntary organizations, and by official institutions. A monograph summarizing the early history of such studies as seen from an institutional perspective, *Humans to Mars: Fifty Years of Mission Planning*, *1950-2000*, was recently completed [Portree 2001] and can be downloaded from http://history.nasa.gov/monograph21/Front%20Matter.pdf . However, the internet has more recently become the primary vehicle for organizing these efforts and for disseminating their resulting reports. The Mars Society, http://www.marssociety.org/ , a voluntary association formed in 1997 to promote manned Mars missions, has accumulated a body of Mars-relevant technical papers from its annual conventions and from its online project collaborations. Another online collaboration more focused on purely engineering issues was formed by the MarsFoundation group; it is organized around the following website: http://www.marshome.org/ .

In the course of searching for technical literature relevant to this thesis project, a technical report was found that resulted from a NASA-funded study of integrating a nuclear reactor into a manned rover for Mars missions [El-genk and Morley 1991]. It

was carried out at the University of New Mexico's Institute for Space Nuclear Power Studies (as a thesis effort), but is available at the following NASA website:
http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19910018625_1991018625.pdf .

This study rejected employing the Open Brayton Cycle for thermal conversion based on its estimate that the high compression ratios needed with martian air would require too massive a complement of axial-flow compressors and turbines. However, it failed to consider using radial flow compressors and turbines which typically have a much greater pressure ratio capability per stage. Since in addition that study did not model the nonlinear thermodynamic behavior of the martian air and it also minimized its deviations from the SP100 design architecture used as the starting point for its reactor design, it seems appropriate to set its conclusions aside for the present thesis project.

CHAPTER 2
2 INFORMATION CONCERNING MARS

Since mobile nuclear engines designed herein will function in the martian environment, it is appropriate to review relevant facts known about Mars. Most of the included facts derive from research organized or supported by NASA, although some derived from the European Space Agency (ESA). Direct sources of this information included scientific journal publications, internet sites of university researchers participating in Mars research, the internet sites of NASA and its affiliated laboratories (particularly JPL) and its other contractors, and the internet sites of the ESA.

Facts were chosen for inclusion here based on their relevance to this design project. All information concerning the martian air is relevant for the design of air-breathing engines. Other important information concerns the martian climate, topography, water, surface mineral composition, and the radiation environment.

2.1 Bulk Parameters

Figure 2-1 visually compares Earth and Mars by showing them in the same scale. Table 2-1 reveals Mars and Earth have about the same dry land surface area but Earth also has liquid oceans while Mars has none. Mars' topography has significantly more vertical relief than Earth, with taller mountains and deeper valleys. Estimates of Mars' mass, developed during the 1800s based on astronomical observations of the orbits of Mars' two moons, Phobos and Deimos, show that Mars' mean density is less than Earth's but more than typical densities of Earth's surface minerals. This has been interpreted as evidence that the hypothesized liquid iron cores of the two planets differ in their fractions of total planetary mass. Mars' surface gravity is about 38% of Earth's surface gravity while Mars' escape velocity is about 45 % of Earth's. The mean length of a martian day, also known as a "sol", is 24 hours 39 minutes 35 seconds, while the duration of a martian year is 668.595 sols. Since Mars' equator is tilted with respect to its orbital plane by about the same angle as Earth's, the northern and southern hemispheres of both planets undergo similar summer, autumn, winter, and spring seasonal changes. Polar regions of both planets endure continuous midsummer sunlight and continuous midwinter darkness. Sunlight reaching the top of Mars' atmosphere is about 43% as intense as sunlight at the top of Earth's atmosphere, but a larger average fraction of it penetrates through the usually clear martian atmosphere than through Earth's frequently cloudy skies.

Most of what is known about Mars results from unmanned spacecraft visiting that planet. Table 2-2 lists the 38 unmanned spacecraft launched prior to 2008 on missions to Mars. Seven of these carried both an orbiter and a separate lander, namely, Mars2, Mars3, Mars6, Viking1, Viking2, Phobos2, and Mars Express/Beagle2. Additionally, the Mars Polar Lander spacecraft was also composite, carrying a soil-penetrating lander. Thus, the 38 launches carried a total of 45 vehicles towards Mars. Of these, 19 were partially or fully successful while 26 failed, returning no information whatsoever.

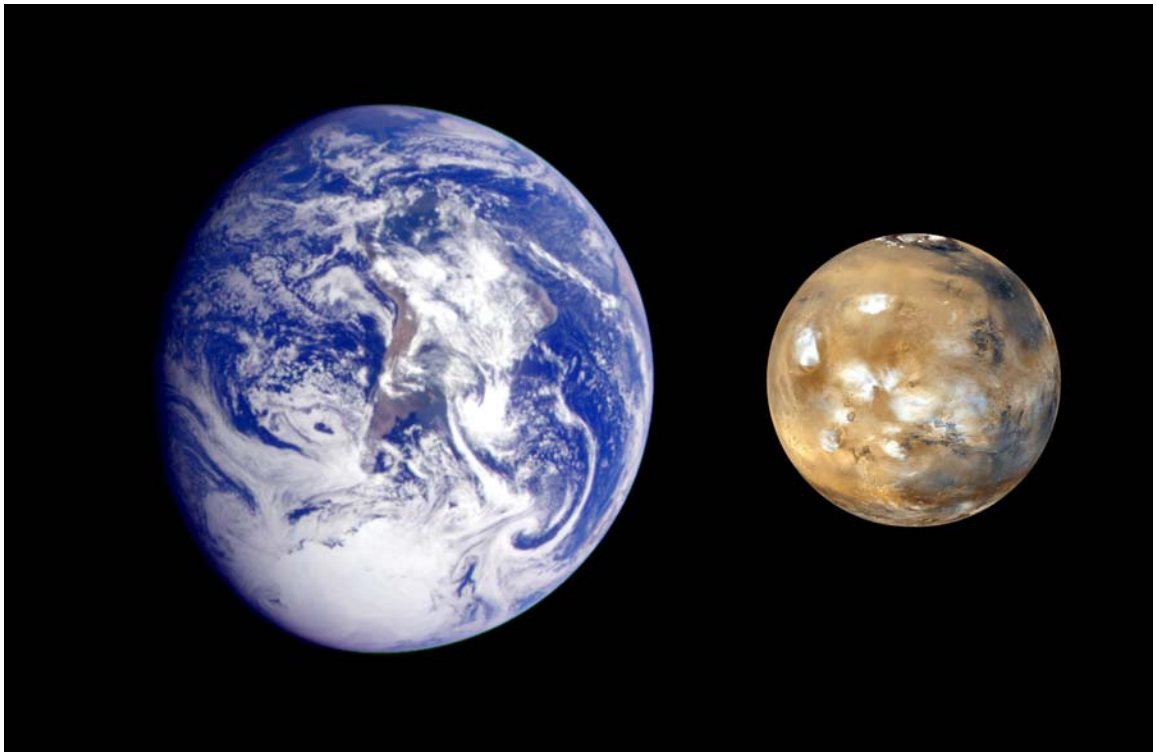


Figure 2-1: Scaled Photo Images of Earth and Mars (Images Courtesy NASA/JPL)

Table 2-1: Comparison of Selected Bulk Parameters for Earth and Mars

Type	Parameter	Earth	Mars	M/E Ratio
Size and Shape	Equatorial Circumference (km)	40,075	21,344	0.5326
	Mean Equatorial Radius (km)	6,378.14	3,397	0.5326
	Mean Polar Radius (km)	6,356.8	3,375	0.5309
	Surface Topographic Altitude Range (km)	20	36	1.8
	Dry Land Surface Area (km^2)	147,919,100	144,100,000	0.9742
	Liquid Surface Area (km^2)	362,146,600	0	0
	Total Surface Area (km^2)	510,065,700	144,100,000	0.2825
	Volume (km^3)	$1.0832*10^{12}$	$1.6314*10^{11}$	0.1506
Satellites	Number of Natural Moons	1	2	2
Mass	Mass (kg)	$5.9737*10^{27}$	$6.4185*10^{26}$	0.1074
	Mean Density (kg/m^3)	5,515	3,934	0.7144
	Equatorial Surface Gravity (m/s^2)	9.766	3.693	0.3781
	Surface Escape Velocity (km/s)	11.19	5.03	0.4495
Rotation	Mean Length of day (hrs)	24.0000	24.6597	1.02749
	Obliquity (Equator's Inclination to Orbit, degrees)	23.45	25.19	1.0742
Orbit	Length of Year (in Earth-days)	365.242	686.973	1.8809
	Orbital Eccentricity	0.0167	0.0935	5.599
	Inclination to Ecliptic Plane (degrees)	0	1.849736	---
	Mean Distance from Sun (km)	149,597,890	227,936,640	1.5237
Thermal	Solar Irradiance above atmosphere (W/m^2)	1367.6	589.2	0.4308
	Blackbody Temperature (Kelvins)	254.3	210.1	0.83
	Mean Surface Temperature (Kelvins)	288	215	0.75

Table 2-2: History of Unmanned Spacecraft Missions to Mars (Courtesy NASA)

Launch Date	Name	Country	Result	Reason
1960	Korabl 4	USSR (flyby)	Failure	Didn't reach Earth orbit
1960	Korabl 5	USSR (flyby)	Failure	Didn't reach Earth orbit
1962	Korabl 11	USSR (flyby)	Failure	Earth orbit only; spacecraft broke apart
1962	Mars 1	USSR (flyby)	Failure	Radio Failed
1962	Korabl 13	USSR (flyby)	Failure	Earth orbit only; spacecraft broke apart
1964	Mariner 3	US (flyby)	Failure	Shroud failed to jettison
1964	Mariner 4	US (flyby)	Success	Returned 21 images
1964	Zond 2	USSR (flyby)	Failure	Radio failed
1969	Mars 1969A	USSR	Failure	Launch vehicle failure
1969	Mars 1969B	USSR	Failure	Launch vehicle failure
1969	Mariner 6	US (flyby)	Success	Returned 75 images
1969	Mariner 7	US (flyby)	Success	Returned 126 images
1971	Mariner 8	US	Failure	Launch failure
1971	Kosmos 419	USSR	Failure	Achieved Earth orbit only
1971	Mars 2 Orbiter/Lander	USSR	Failure /Failure	Orbiter arrived, but no useful data and Lander destroyed
1971	Mars 3 Orbiter/Lander	USSR	Success /?Success	Orbiter obtained approximately 8 months of data and lander landed safely, but only 20 seconds of data
1971	Mariner 9	US	Success	Returned 7,329 images
1973	Mars 4	USSR	Failure	Flew past Mars
1973	Mars 5	USSR	Success	Returned 60 images; only lasted 9 days
1973	Mars 6 Orbiter/Lander	USSR	Success /Failure	Occultation experiment produced data and Lander failure on descent
1973	Mars 7 Lander	USSR	Failure	Missed planet; now in solar orbit.
1975	Viking 1 Orbiter/Lander	US	Success /Success	Identical twin craft returned 16,000 images, located landing sites for Landers and made first Mars landings that were followed by sustained operation. Made extensive Mars' air and soil measurements, also performed microbial life experiments.
1975	Viking 2 Orbiter/Lander	US	Success /Success	
1988	Phobos 1 Orbiter	USSR	Failure	Lost en route to Mars
1988	Phobos 2 Orbiter/Lander	USSR	Failure /Failure	Lost near Phobos
1992	Mars Observer	US	Failure	Lost prior to Mars arrival
1996	Mars Global Surveyor	US	Success	More than 240,000 images through October 2006; Extensive topographic maps via laser altimeter.
1996	Mars 96	Russia	Failure	Launch vehicle failure
1996	Mars Pathfinder	US	Success	Technology experiments; local mineral assays.
1998	Nozomi	Japan	Failure	No orbit insertion; fuel problems
1998	MarsClimate Orbiter	US	Failure	Lost on arrival; human errors in navigation
1999	MarsPolarLander	US	Failure	Lost on arrival during atmospheric entry
	Deep Space 2 Probes (2)	US	Failure	Lost on arrival during atmospheric entry (carried on Mars Polar Lander)
2001	Mars Odyssey	US	Success	High resolution infrared, gamma ray, neutron spectrometers
2003	MarsExpressOrbiter/ Beagle2Lander	ESA	Success /Failure	Orbiter imaging Mars in detail and using first subsurface penetrating radar, but lander lost on arrival
2003	MarsExploration Rover - Spirit	US	Success	Over 70,000 images; rock analyses; traveled kilometers
2003	MarsExpl.Rover - Opportunity	US	Success	Over 58,000 images; rock analyses; traveled kilometers
2005	MarsReconnaissance Orbiter	US	Success	First very high resolution images of Mars' surface with pixel size 29 centimeters were obtained during September 2006.
2007	PhoenixMars Lander	US	Success	Subsurface water-ice excavated with direct chemical analysis of it and ground minerals. Atomic force microscope imaged dust grains in soil with submicron size resolution.

Mariner 4, the first successful Mars flyby, returned the earliest close-up pictures of Mars. Some of these showed lunar-type craters touched with white evening frost. Radio signal occultation gave the first evidence that the martian atmosphere is thin compared with Earth's air. The Mariner 6 and 7 flybys provided the first close-up pictures of the thick south polar icecap. Mariner 9, the first successful orbiter, arrived at Mars when a global dust storm was obscuring most of the planet. A month later after the dust had cleared Mariner 9 initiated photo-mapping Mars' surface which it completed within a year. In addition to impact craters its photos revealed gigantic volcanoes of which one (Olympus Mons) stands 27 kilometers above the mean surface, a gigantic canyon (Valles Marineris) plunging to 11 kilometer depths and stretching 4,000 kilometers across Mars' surface, an equally deep impact basin the size of Australia (Hellas Planitia), and systems of apparent riverbeds carved in the arid landscape. The twin Viking missions each had an orbiter and a lander. The orbiters worked for 2 years and 4 years respectively before failing while the two landers worked for 4 years and 6 years respectively. Each provided many photos. Each lander analyzed the composition of soil within reach and performed biological experiments designed to detect microorganisms with unexpected results interpreted as negative for life. The landers also studied the martian air, each performing assays of air composition and establishing records of time-varying pressure, temperature and wind speed at two elevations on their masts.

2.2 Atmosphere

Martian air pressure measurements by the Viking landers as presented by Prof. James Tillman on his website, http://www-k12.atmos.washington.edu/k12/resources/mars_data-information/mars_overview.html, are included below as Figure 2-2. Pressures in millibars (1 bar=10^5 Pascals) are plotted versus time in sols, with arrows delimiting *martian years*. Figure 2-3 overlays successive years to show annual repeatability.

Measured pressures at the Viking sites ranged from 690 Pa to 1010 Pa. The global pressure variation must be larger, perhaps 350 Pa to 2000 Pa for a ±7 km altitude range.
It follows from martian surface pressure, area, and gravity that the total mass of the entire martian atmosphere is about (800 Pa)(1.441*10^{14} m^2)/(3.693 m/s^2)≈3*10^{16} kg. However, that estimate is inherently approximate since Mars' atmospheric mass changes substantially over time. The large biennial pressure fluctuations of Figures 2-2 and 2-3 are caused by imbalances between CO_2 gas freezing out as frost at the planet's dark pole and CO_2 frost subliming into gas at its sunlit pole.

Typical measured wind speeds at Viking lander sites ranged from 2 to 7 m/s in summer and winter, 5 to 10 m/s in spring and autumn, but during dust storms rose to the 17 to 30 m/s range. Martian air density varies with its pressure and its temperature, but a typical martian air density is about 0.020 kg/m^3. A typical density of Earth's air is about 1.2 kg/m^3. The mean molecular weight of the martian air as determined by the Viking landers is 43.34 g/mole, and its chemical composition is shown in Table 2-3. Typical night-to-day air temperature ***changes*** ranged from 55 to 60 Kelvins.

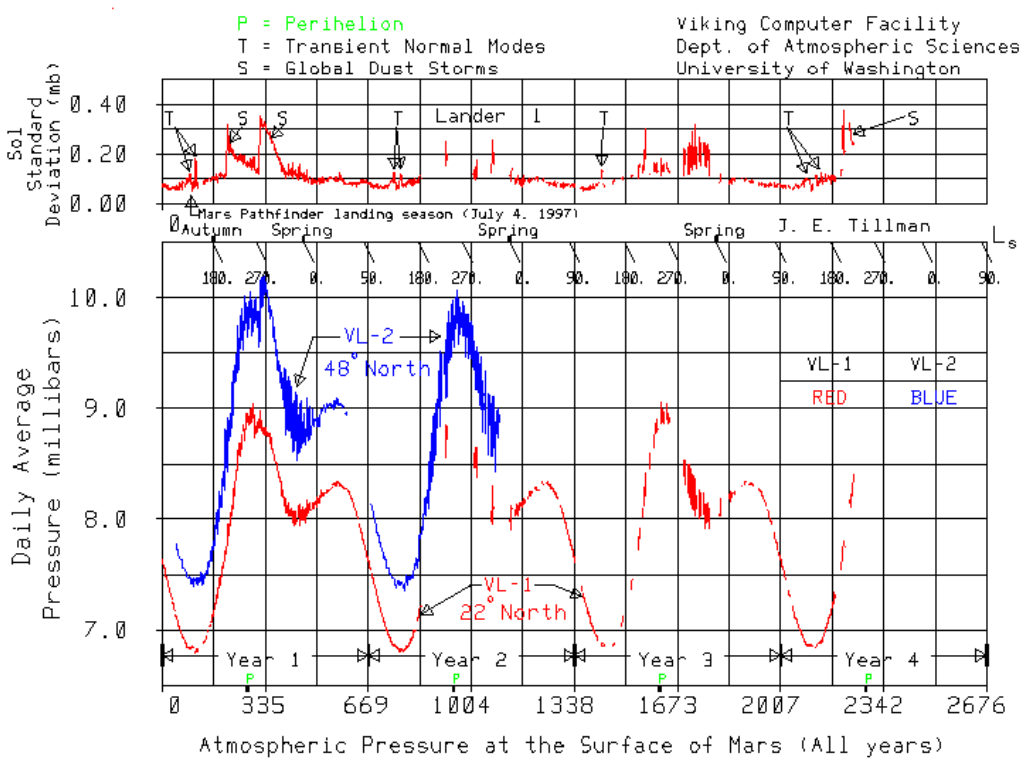


Figure 2-2: Martian Air Pressure Records of Viking 1 & 2 Landers (Courtesy J. Tillman)

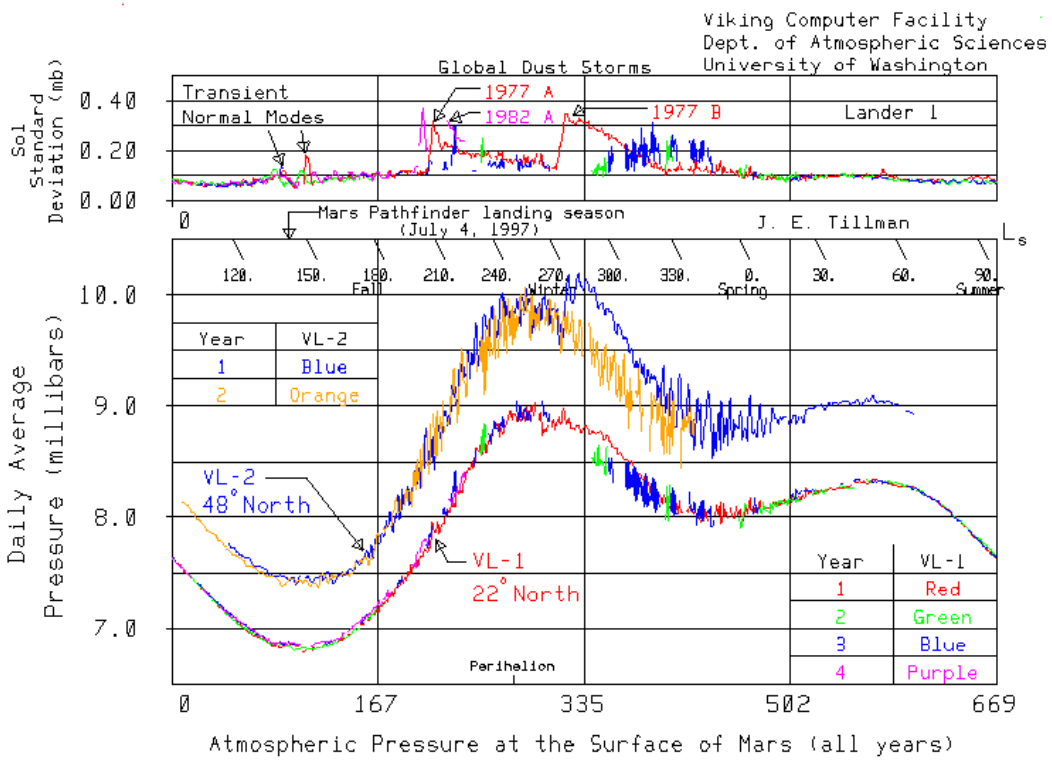


Figure 2-3: Successive Martian Year Pressure Records Overlay (Courtesy J.Tillman)

Table 2-3: Martian Air Composition (by volume)

Martian Air Chemical Component	Volume Per Cent
Carbon Dioxide (CO_2)	95.32
Nitrogen (N_2)	2.7
Argon (Ar)	1.6
Oxygen (O_2)	0.13
Carbon Monoxide (CO)	0.08
Water Vapor (H_2O)	0.021
Nitrogen Oxide (NO)	0.010
Neon (Ne)	0.00025
Krypton (Kr)	0.00003
Xenon (Xe)	0.000008

Viking atmospheric composition and temperature data have been used to estimate an approximate e-folding scale height of 11.1 km for the martian atmosphere. For comparison, the corresponding approximate e-folding scale height of Earth's atmosphere is 6.0 km. Because Mars' air density falls off more slowly with altitude than Earth's, the two planets have identical air densities at an altitude of roughly 55 km. Mars' air is denser than Earth's air at higher altitudes where much of the kinetic energy of an arriving interplanetary spacecraft must be dissipated by atmospheric entry aerobraking.

After a 20 year hiatus, exploration of Mars by unmanned probes resumed with the 1996 Mars Global Surveyor (MGS) orbiter spacecraft. This was the first spacecraft to rely on controlled aerobraking instead of rockets to transition from an extremely elliptical capture orbit to a circular low altitude orbit suitable for surface mapping studies. The MGS was equipped with the Mars Orbiter Laser Altimeter (MOLA), an instrument able to measure localized elevations of Mars' surface under the orbit by beaming ten infrared laser pulses per second at the ground beneath the spacecraft and timing the reflections from the ground. Its 8 nanosecond laser pulses provided instrument resolution of surface altitudes to a stated precision of 0.375 meters while the laser spot size on the ground was 130 meters in diameter. Since the MGS orbit was nearly polar, successive orbits scanned most of Mars' surface as the planet rotated beneath the MGS orbit plane.

2.3 Topography

A preliminary color-coded topographic global map of Mars' surface was released soon after the number of accumulated MOLA surface altitude measurements reached one million, and it is reproduced as Figure 2-4. It presents information in three parts. Its lower rectangular Mercator projection covers latitudes ranging from -70º to +70º. The upper left cap covers latitudes from -90º to -70º in the south polar region while the upper right covers latitudes from +70º to +90º in the north polar region. At that early release time, elevation data were still missing for a small region around the south pole never under the MGS orbit, but had been filled in for the north pole by specially timed maneuvers deliberately tilting the spacecraft from its usual nadir orientation.

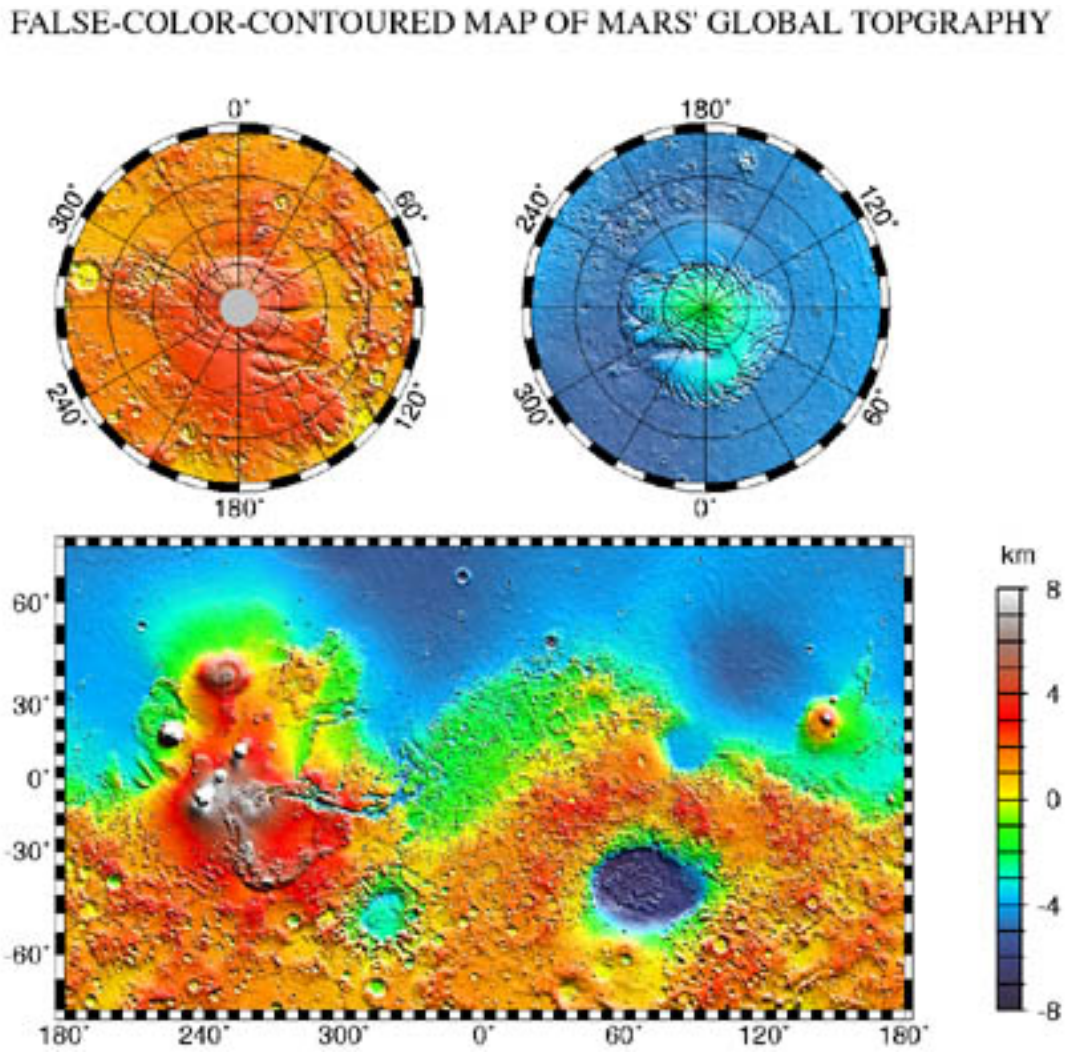


Figure 2-4: Initial Topographic Map of Mars From MGS MOLA Measurements

The elevation scale in Figure 2-4 ranging ±8 km resolves elevations of interest for much of Mars' surface but fails to delimit tall volcanic mountains (such as Olympus Mons at 225ºE 20ºN), or the deepest spots either in the Valles Marineris canyon (which extends from 270ºE 5ºS to about 330ºE 5ºS) or in the Hellas impact basin (at about 70ºE 40ºS).

Figure 2-4 shows a planetary asymmetry between Mars' low elevation northern hemisphere and its southern hemisphere which averages about 5 kilometer higher elevation. This was highlighted by a diagram published in the journal Science in 1999 [Smith et al 1999], reproduced here as Figure 2-5. It plots the elevation profile along a swath following Mars' prime meridian, with the north pole at the left and the south pole at the right:

Apparent dry river beds which MOLA data showed are deeper than previous photo-based estimates descend from south to north and disappear when they reach lower regions colored blue in Figure 2-4. An apparent dry river bed example appears in Figure 2-6, a mosaic of photos from a different MGS instrument, the Mars Orbital Camera (MOC) developed and operated by Malin Space Science Systems. Figure 2-6 covers longitudes from 90ºW(270ºE) to 45ºW(315ºE) and latitudes from 0º to 30ºN..

A second asymmetry is that the southern highlands are densely pocked with impact craters while the low northern plains are relatively free of impact craters. Mars' low northern plains are among the most level and smooth surfaces in the solar system, rivaling Earth's oceans. Some of the northern impact craters which do exist show evidence of large outflows of "fluid ejecta" occurring when the craters formed. This is shown for example in the Figure 2-7 Viking Orbiter context image which is expanded in the Figure 2-8 MGS/MOC image. The ejecta resembles dried mudflows. The ground may have contained significant fractions of frozen volatiles which became fluid upon impact.

Because of such facts, there are ongoing scientific debates over whether Mars' northern plains were once the site of an ancient ocean and even further over whether frozen ocean remnants might still remain buried beneath a surface layer of soil. Some published papers have claimed photographic evidence for ancient ocean shorelines while others have debunked them [Parker and Banerdt 1999], [Head et al 1999], [Ivanov and Head 1999], [Barlow 1999].

Regardless how northern ocean controversies are eventually resolved, MGS MOLA data found large quantities of martian water in polar icecaps. MGS radiometer data showed the northern icecap's summer surface temperature hovers at the freezing point of water after its 2 meter thick surface layer of CO_2 frost deposited during the preceding winter (also measured by MOLA) first sublimes away. This and other data have convinced scientists that the northern icecap is mostly water ice. A 1998 paper by the MOLA team in the journal Science [Zuber et al. 1998] described the northern polar cap as rising a maximum of 3 kilometers above the surrounding flat surface and containing a volume calculated from MOLA data as between $1.2 * 10^6$ and $1.7 * 10^6$ km^3, depending on which

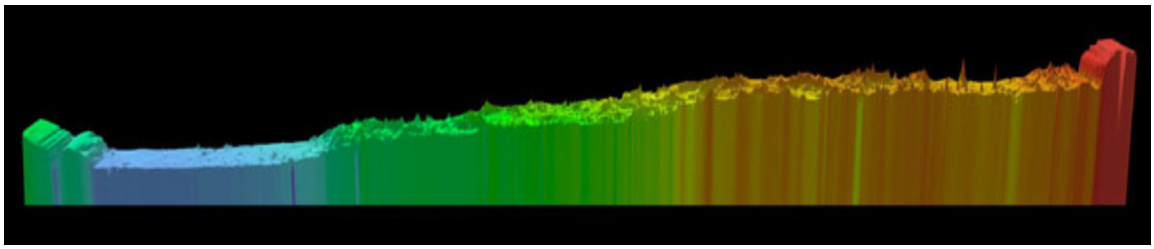


Figure 2-5: Elevations On Mars' Prime Meridian (Courtesy MOLA Science Team)

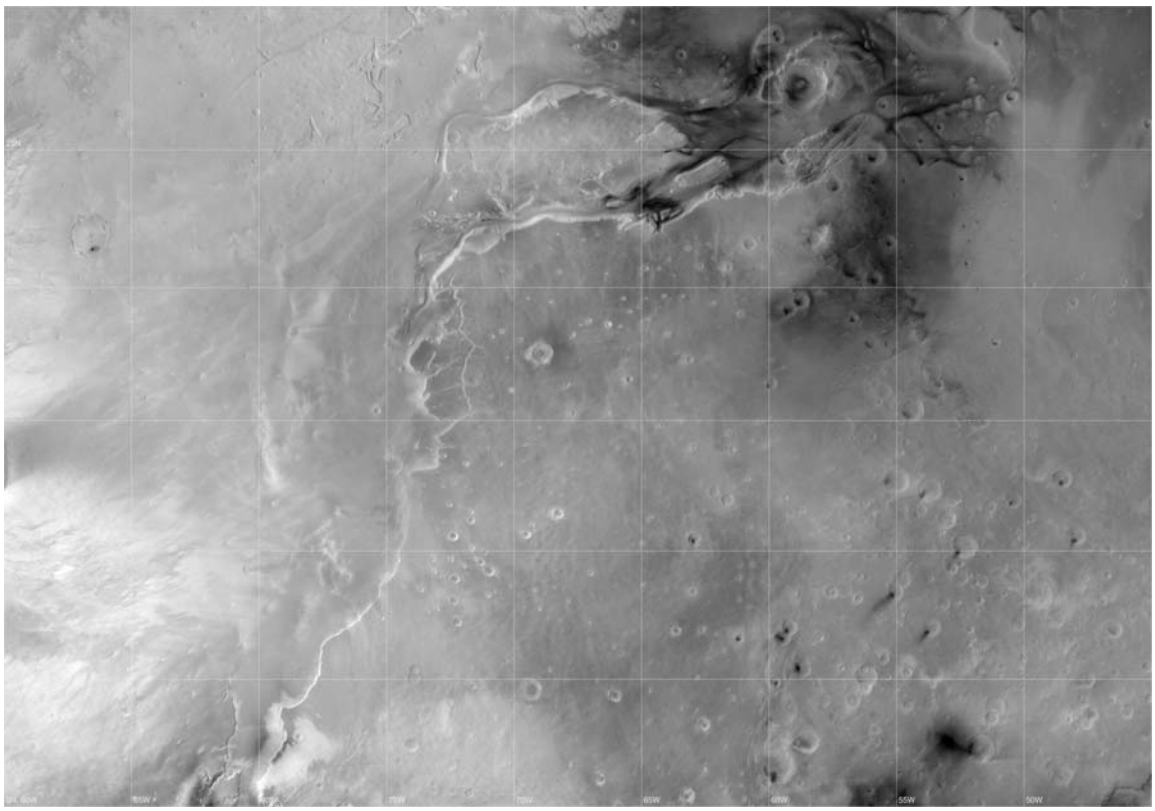


Figure 2-6: An Apparent Dry River Bed On Mars (Courtesy NASA/JPL/MSSS)

.

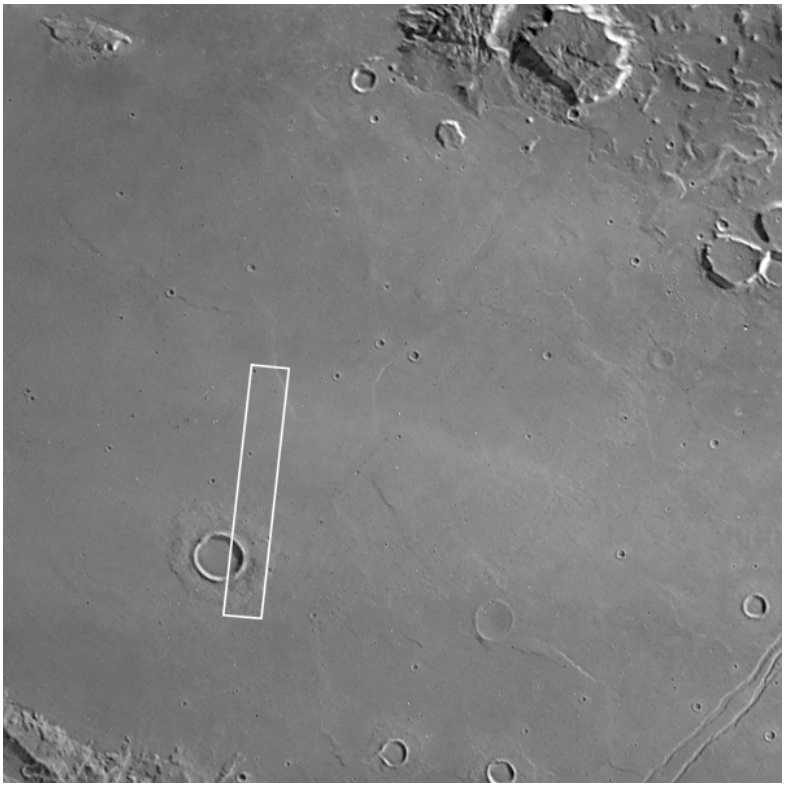


Figure 2-7: Viking Orbiter Photo of Crater With Fluidized Ejecta

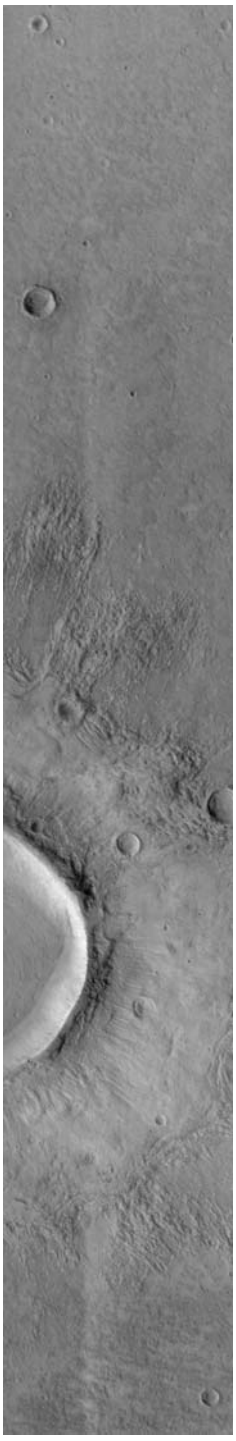


Figure 2-8: MGS/MOC Closeup

assumptions are made about the shape of the hidden bottom of the ice cap, i.e., how Mars' lithosphere is locally depressed under the concentrated weight. If the north icecap is pure water ice its mass would be between 10^{18} and $1.5 * 10^{18}$ kg, or between 35 and 50 times the mass of the entire martian atmosphere. To better understand this amount of water it is instructive to divide these limiting volume estimates by Mars' surface area, $1.441 * 10^{8}$ km^2. The resulting quotients show north polar ice alone could cover the entire planet to a uniform water ice depth between 8.7 and 12.3 meters (i.e., between 28.5 and 40.3 feet). MOLA estimates of the larger southern polar cap's volume, if added to the northern cap, increases the calculated global depth of uniformly redistributed polar ice to 30 meters (i.e., about 100 feet). The southern polar cap's composition is less certain. Its thermal behavior indicates a significant fraction of it may be CO_2 frost in addition to its water ice content. A counter argument asserts that solid CO_2 is not strong enough to structurally support the ice thickness which MOLA data show the southern ice cap actually has. However, if the southern polar ice deposits were mostly CO_2 then if they were ever to entirely sublime into a gas Mars' surface air pressure might surpass Earth's.

MOLA data yields vivid computer-generated images, e.g., Olympus Mons as shown in Figure 2-9. Olympus Mons' central edifice reaches 27 kilometers above the mean surface. It is 550 km in width (roughly the size of Arizona) and has a caldera 85 km across and up to 3 km deep. Its outer edge is defined by a steep escarpment 6 km tall.

While the MGS spacecraft continued to provide large volumes of MOLA, MOC, and other data each year, precise doppler radio tracking over time of the MGS spacecraft also provided a map of Martian gravitational anomalies, i.e., departures from a purely spherical gravity field. The resulting gravity anomaly maps were published in the journal Science in October 1999 [Smith et al]. In turn, they enabled small non-spherical corrections to be made to the "sea level" type of surface altitude from which topographic maps are referenced. The United States Geologic Survey (USGS) took on this task along with the reinterpretation and integration of the by then vastly larger quantity of accumulated MOLA data (i.e., 600 million elevation measurements). The result was the 2003 USGS publication of corrected topographic maps of Mars similar to the Figure 2-4 maps but with such a finely detailed scale (i.e., 64 pixels per degree providing <1 km horizontal resolution) that they cannot properly fit on the pages here. They are electronically available to download directly from the USGS as "pdf" files (at internet web site http://pubs.usgs.gov/imap/i2782/) or they can be purchased from USGS on large paper sheets. Even higher resolution MOLA information, down to the 130 meter diameter of the laser spot on Mars' surface, is electronically available as "gridded data sets" from NASA's Planetary Data Node at the web site address, http://pds-geosciences.wustl.edu/missions/mgs/megdr.html, along with data format explanations.

Continuing MOLA data has been used to study topographic ***changes*****,** which are significant for the polar ice caps. Investigators discovered that for three martian summers in a row, identifiable CO_2 frost deposits near the south pole shrank to smaller sizes each summer than the previous year's minimum, signifying an underlying trend towards thinning of the frost. This suggests Mars is now in a global warming phase.

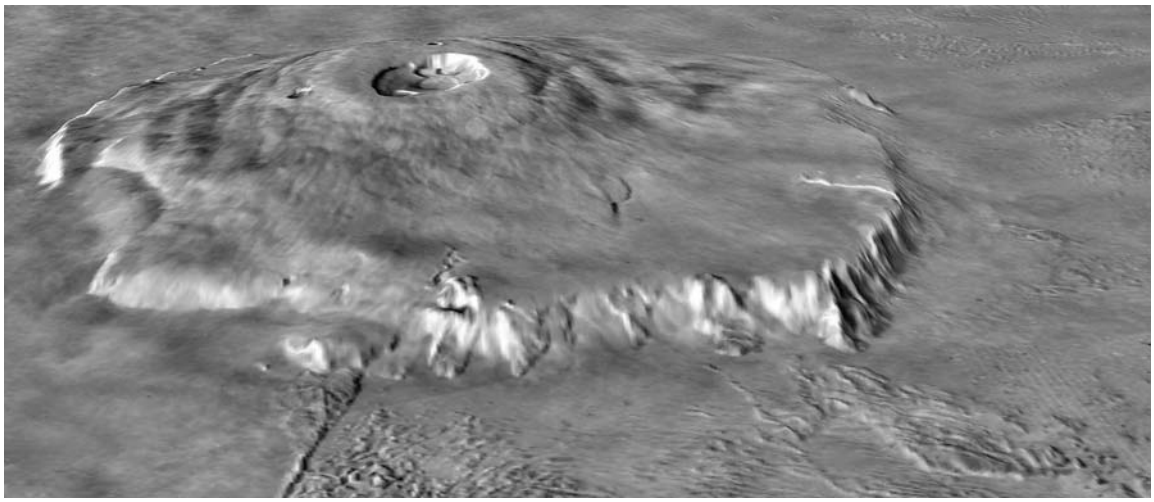


Figure 2-9: Olympus Mons with Vertical Scale Exaggerated 10 Times Horizontal (Courtesy NASA/JPL)

The MGS was last heard from on November 2, 2006, when its telemetry reported solar array trouble. A NASA review board determined that human error had caused a fatal loss of power. In retrospect, the MGS was the most productive of all equipment sent to Mars, leaving us more than 240,000 high resolution images in addition to much other data.

2.4 Surface Composition

The 1996 Pathfinder mission was originally conceived as an engineering technology experiment. It tested the first complete reliance on aerobraking to slow down from interplanetary speeds without first using a retrorocket to enter Mars orbit. It tested a new method for landing after a parachute descent by using terminal air bags instead of a touchdown rocket. It also tested the first use of a mobile wheeled robot, named Sojourner. While much of its mission return concerned technology development, the alpha proton spectrometer instrument carried by Sojourner provided compositional information about local rocks and soil near the landing site. Preliminary results for compositions of samples, along with a comparison with Earth crustal compositions, were given on the Pathfinder web site. They are reproduced here as Table 2-4 and Figure 2-10. Table 2-4 lists the main minerals and their weight composition fractions. Figure 2-10 plots calcium vs. iron contents for different samples with each element normalized to the sample's silicon content for a variety of sample sources from Mars and from Earth; it shows that martian samples tend to contain more iron than terrestrial samples. In summary, Mars surface samples are similar to typical Earth samples in that they are about half silica (SiO_2) by weight. Iron oxide, the second most abundant component, is about twice as abundant on Mars' surface as it is on Earth. The other major components of Earth's surface such as Al_2O_3 and CaO are also major components of Mars' surface.

The 2001 Mars Odyssey orbiter included a gamma ray spectrometer from which the inferred abundances of several elements in the top two meters of soil were separately mapped over the global surface. For example, Figure 2-11 shows global near-surface potassium, which is about twice as abundant on Mars as it is on Earth.

Perhaps of more importance, near-surface hydrogen was also mapped globally. Figure 2-12, reproduced from the 2001 Mars Odyssey website, shows hydrogen results under the plausible assumption that the inferred hydrogen content represents water ice. It reports that near-surface ground material within 30 degrees latitude of either pole is mostly water ice, although water ice is visible from space predominantly in the much smaller polar ice caps. It seems likely that this underlying water ice may exist, covered with a protective layer of windblown dust, but perhaps instead the hydrogen may be present in other mineral forms (e.g., gypsum). Figure 2-12 also reports regions on Mars' ***equator*** where the inferred ground water mass fraction in the top two meters of soil is in the four to eight per cent range (e.g., on the prime meridian). This may be important for manned Mars missions since equatorial locations have other attractive features.

Table 2-4: Mars PathfinderAnalysis of Martian Samples by the Alpha Proton Spectrometer:

Mars (weight percent)			Earth (weight percent)		
Ordered most to least	A-3, Rock "Barnacle Bill"	A-5, Soil	Continental Crust		Oceanic Crust
			Average	Sediments	
SiO_2	55.0	43.8	60.2	50.0	50.7
Al_2O_3	12.4	10.1	15.2	13.0	15.6
FeO	12.7	17.5	6.05	5.5	9.9
FeO/MnO	14.1	29.2	-	-	-
CaO	4.6	5.3	5.5	8.4	11.4
MgO	3.1	8.6	3.1	3.1	7.7
K_2O*	1.4	0.7	2.9	2.0	0.17
TiO_2	0.7	0.7	0.7	0.7	1.5
MnO*	0.9	0.6	0.1	0.1	0.16

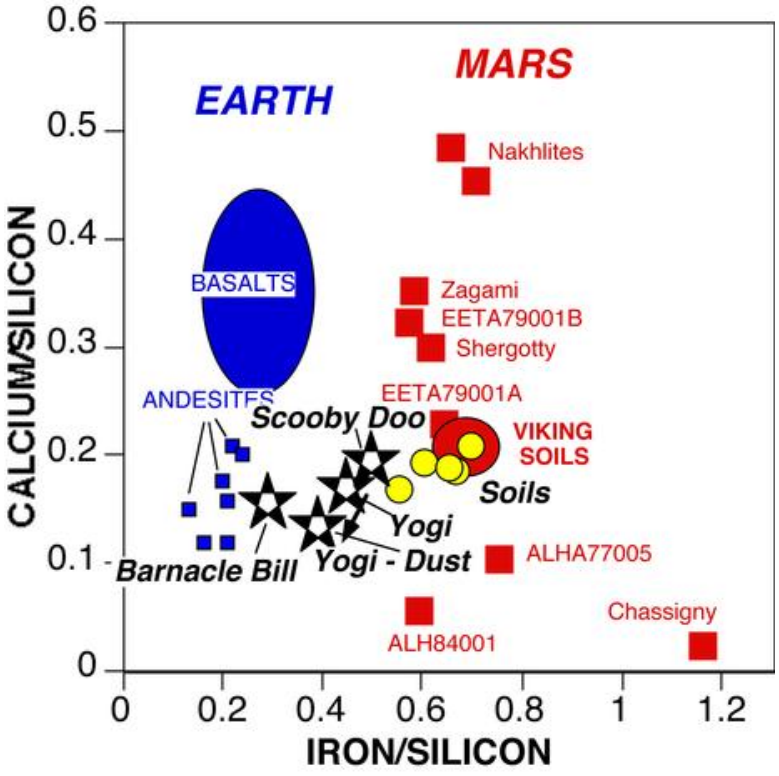


Figure 2-10: Mars/Earth Surface Composition Comparison Courtesy NASA)

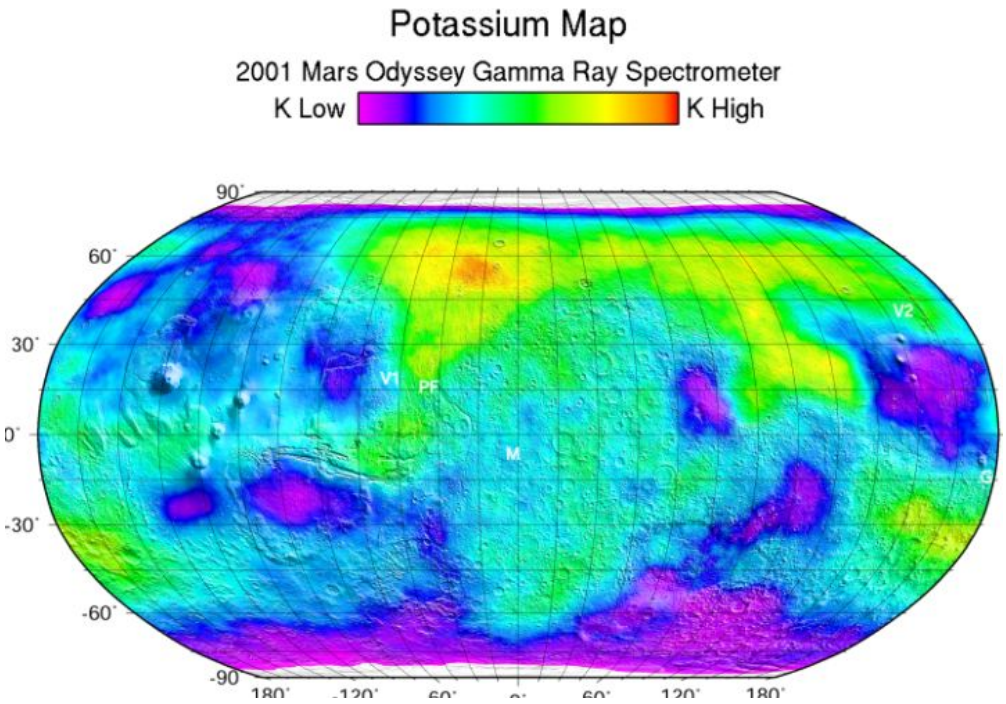


Figure 2-11: Near-Surface Mars Potassium Content, from 2001 Odyssey Spacecraft's Gamma Ray Spectrometer (Courtesy NASA/JPL)

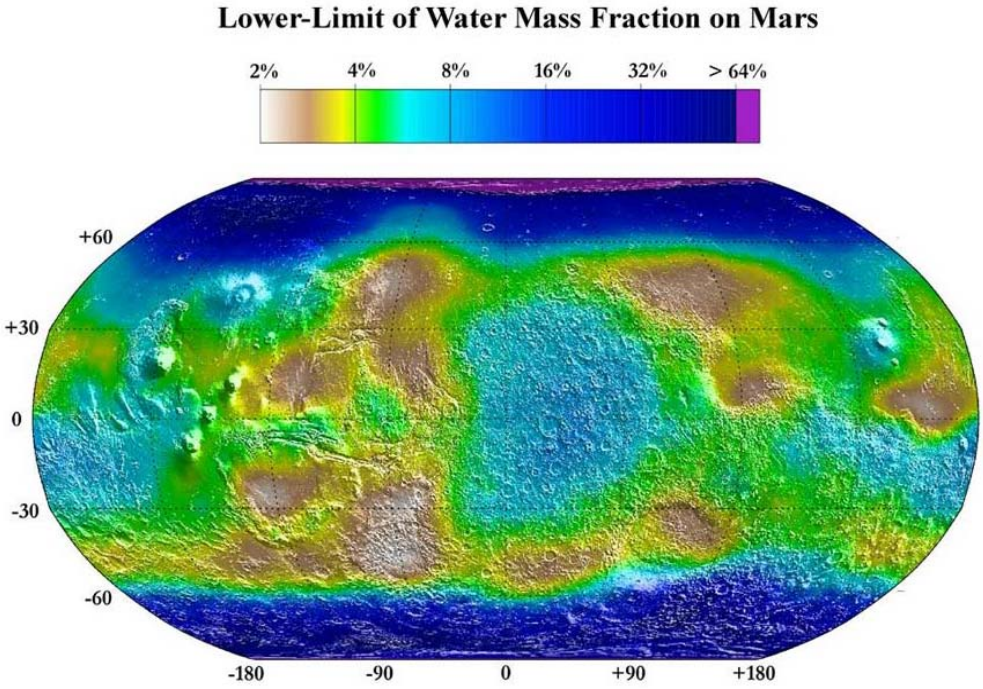


Figure 2-12: Near-Surface Mars Water Content Inferred From 2001 Odyssey Spacecraft's Gamma Ray Spectrometer Measurements (Courtesy NASA/JPL)

The 2001 Mars Odyssey mission featured a Thermal Emission Imaging System (THEMIS) providing imaging in 9 infrared and 5 visible spectral bands with 18 meters per pixel ground resolution. This provided not only comprehensive temperature mapping of the ground surfaces over time showing temperatures ranging globally from 140 Kelvins to 300 Kelvins, but also separate signature identification of many different soil consistencies and surface mineral compositions. Its results are summarized on-line at the THEMIS instrument web site at Arizona State University, http://themis.asu.edu/. A mosaic of THEMIS images of Valles Marineris appears as Figure 2-13.

The Figure 2-14 photo, taken in February 2005 (late summer in the northern hemisphere) by the Mars Express orbiter of the European Space Agency (ESA) shows a 12 km diameter smooth circular "lake" 2 km below the rim of a 35 km diameter crater with no name located at 103ºE,70.5ºN. By this time of year, nearby CO_2 frost had sublimed to gas so this is presumed to be pure water-ice, part of Mars' surface composition.

The Phoenix Mars Lander which on 25 May 2008 landed at a latitude of 68.16ºN, about 1200 km from Mars' north pole, discovered that beneath several centimeters of dusty soil lies a smooth horizontal surface of impenetrable water-ice in nearly pure form. Its depth is unknown.

This lander also has imaged samples of the soil particles covering the ice using an atomic force microscope of unprecedented resolution. The typical particle size found is about one micron in diameter. Since the local soil particles are thought likely to have arrived as windblown dust, this may characterize the typical particle size in martian dust storms.

2.5 Ionizing Radiation

The 2001 Mars Odyssey spacecraft also included the Martian Radiation Environment Experiment (MARIE), designed to characterize the radiation exposure which future human visitors to Mars are likely to receive while they are traveling in space near Mars. An appropriate background context is provided by NASA's document, Understanding Space Radiation (FS-2002-10-080-JSC), obtained via links from the website of NASA's Space Radiation Analysis Group, http://srag.jsc.nasa.gov/ :

> **"Crews aboard the space station receive an average of 80 mSv for a six-month stay at solar maximum (the time period with the maximum number of sunspots and a maximum solar magnetic field to deflect the particles) and an average of 160 mSv for a six-month stay at solar minimum (the period with the minimum number of sunspots and a minimum solar magnetic field). On Earth, we receive an average of two mSv every year from background radiation alone."**

Typical chronic radiation doses received by astronauts in Earth orbit are higher than US standards for terrestrial radiation workers, but astronauts are not legally classified as

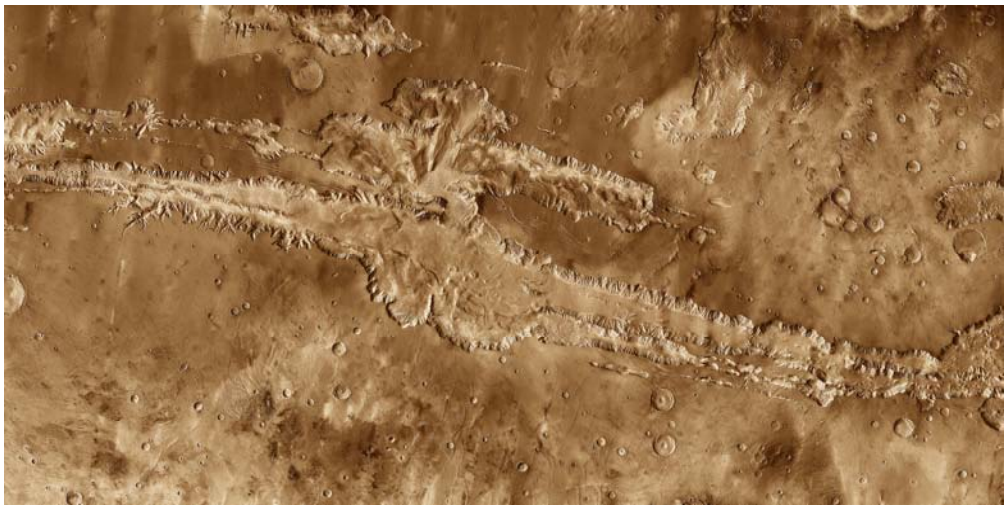


Figure 2-13: THEMIS mosaic image of Valles Marineris (Courtesy NASA/JPL/ASU)

Figure 2-14: Water-Ice Lake in Mars Crater 1100 km From North Pole (Courtesy ESA)

terrestrial radiation workers. The MARIE experiment measured radiation levels in low Mars orbit and found they average about 2.5 times the average radiation levels in low earth orbit aboard the International Space Station (ISS). As reported in space.com [Britt 2003], "NASA sets lifetime limits for astronauts" which "vary by age and gender but generally are between 1 and 3 Sieverts", i.e., between 100 and 300 rem. Extrapolated from MARIE data as measured in low martian orbit to a manned Mars mission duration of "3 years, they represent about 1 Sievert", i.e., about 100 rem. The total radiation dose results from two distinct charged particle sources, a constant galactic cosmic ray (GCR) background and concentrated bursts of solar energetic particles (SEP) which fluctuate enormously based on solar flare events. The lower radiation level at the ISS relative to Mars is due to deflection and trapping of charged particles by Earth's magnetic field. There is no similar magnetic field during interplanetary transit or in low Mars' orbit.

A MARIE web site plot of measurements from Mars orbit appears as Figure 2-15. The data stopped due to instrument failure during a solar event in October 2003. The plot's logarithmic vertical scale shows rad units (i.e., without radiation quality factors applied).

It is noteworthy that most of the total Figure 2-15 integrated radiation dose occurred in brief Solar Proton Event (SPE) bursts. Without SPEs the integrated dose in Mars orbit is only about 8 rad per earth-year. It is also noteworthy that SPEs were observed in Mars orbit by MARIE that were not observed by sensors near Earth, confirming SPEs are emitted directionally by the sun. Radiation levels at Mars' surface are certainly reduced further by atmospheric shielding, but the thin martian air provides less radiation shielding than Earth's air. Although ionizing background radiation on Mars' surface has not yet been measured, it is expected to roughly match levels in low Earth orbit on the ISS.

Additional detailed information about Mars can be examined on the internet and in scientific journal publications. Reviews of archived MGS photo sequences have revealed evidence that liquid water has erupted to the surface during the past seven years, as freshly carved gullies covered with bright deposits resembling dried salt have recently appeared on canyon walls in locations which had been bare in earlier photos. The 2001 Odyssey spacecraft has recently discovered violent carbon dioxide gas eruptions within the southern polar icecap during southern spring and also detected "ghost" radar-type reflections from deep underground far from the poles, suggesting extensive underground water-ice. The Mars Express Orbiter has used a radar instrument to additionally probe for underground water-ice. The Spirit and Opportunity rovers have greatly increased the number of surface samples assayed for chemical composition and other properties and have found mineral evidence that standing and flowing surface water existed for long durations. After an aerobraking phase, the Mars Reconnaissance Observer spacecraft arrived in September 2006 in its engineered low orbit from where its very high resolution camera can generate images with ground surface pixel sizes of 29 centimeters. Although the body of information about Mars is large and growing fast, this review will stop here since this is enough background information about Mars for the present work.

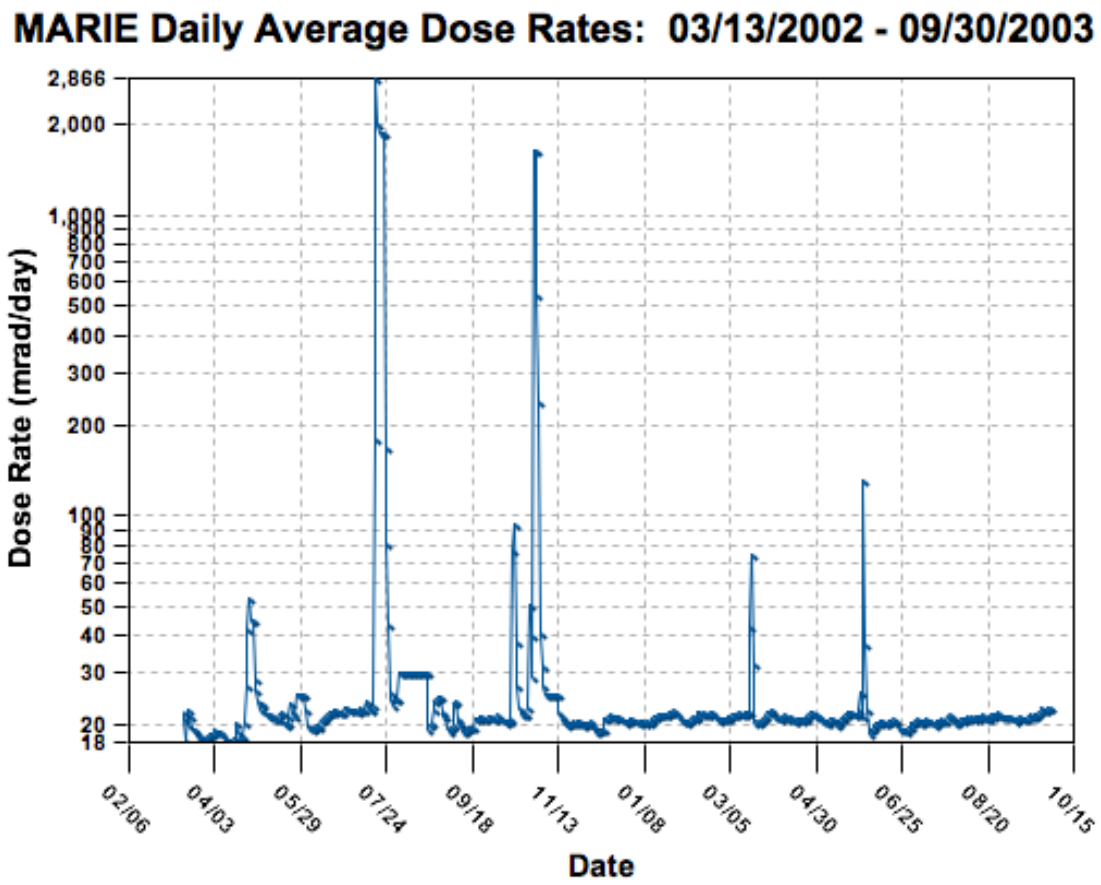


Figure 2-15: MARIE Instrument Measured Radiation in Low Mars Orbit

CHAPTER 3
3 OVERVIEW OF MANNED MARS MISSIONS

"Why design mobile nuclear engines for manned Mars vehicles?"

To answer this basic question we must answer the following questions:

1) Why should humans do anything at all on Mars?
2) Why isn't it sufficient to send tele-operated robots to Mars?
3) What parameters must characterize manned missions to Mars?
4) What problems need resolution to conduct useful manned Mars missions?
5) How do mobile nuclear engines help address those problems?

These subsidiary questions are considered in this chapter. The chapter's purpose is to establish the context underlying the present project in order to help logically choose design requirements and to trade-off conditions. Answers given herein are my own, but the logic behind them is explained.

3.1 Why should humans do anything at all on Mars?

There are two reasons. The longer-term but presently less compelling reason is that ultimately we humans should colonize Mars to enhance prospects for our species' survival along with the survival of other terrestrial life forms which we would bring with us (e.g., green plants). This view has been expressed by many during the past five decades since we first ventured into space. It has recently been endorsed by Dr. Stephen Hawking, one of the world's preeminent physicists.

Mars is the second best place for humans to live in this solar system, second after Earth. Other places are far more hostile. Although Mars is harsh and inhospitable and although humans could not possibly survive on Mars without special equipment, a technical civilization of humans could ***thrive*** on Mars using even today's technologies without requiring continuing support from Earth. Available sunlight under Mars' typically clear skies exceeds average sunlight in typically cloudy regions on Earth where vegetation is lush, so there is enough sunlight on Mars to grow plants. In addition to sunlight, green plants need carbon dioxide and water in order to produce glucose and oxygen gas by photosynthesis. Mars has plenty of water in the form of solid water-ice, both in its visible polar ice caps and buried in high latitude soil. Carbon dioxide, the primary constituent of Mars' atmosphere, is plentiful. There is also enough atmospheric nitrogen on Mars to meet the protein needs of a large living ecosystem. All other chemical elements that would be needed for Earth-life growing in pressurized heated greenhouses are abundant on Mars. In addition, there are no barriers precluding local martian mining and industrial manufacturing of all needed building supplies and equipment, assuming a ready supply of power.

In the long term future, perhaps a century or two from now, surely Mars will be brought within the human economic sphere. The kinetic energy required to propel a trip to Mars is only a small multiple of the energy required to circle the Earth in a jet airplane, but the present ratio of their costs is astronomical. Since energy is such a small part of the total cost it is likely that future technological developments will eventually reduce the cost of space travel by several orders of magnitude while also reducing its hazards.

The nearer term reasons are purely scientific since Mars exploration is not expected to yield ***direct*** benefits to national economies or militaries. The question "Why do anything at all with Mars" can be posed equally well for other scientific areas. "Why should we conduct research in astronomy, biochemistry, paleontology, geology, plasma physics, archeology, nuclear physics, cell biology, high energy particle physics, or other fields?" Basic science and longer term applied science (i.e., engineering) are typically funded by governments, although the nonscientific public may sometimes fail to understand the reasons for such expenditures. Thus, this larger question concerns public policy issues of how we direct government funds into scientific research and engineering development.

Opposition to science comes from various groups. Violent opposition from different religions range from the threats four centuries ago to burn Galileo alive just as his predecessor scientist, Bruno, had been executed, to present-day Wahabist threats to murder professors daring to teach evolutionary biology at Turkish universities. Less violent, but still strong opposition to science by religious fundamentalists is common today in many countries, including the USA. Government spending on science is also opposed by groups who claim that monies should instead be spent to further social goals ranging from income redistribution to environmental preservation to infrastructure renewal to remedial education, along with various other purposes perceived as worthy. Additionally, many people are uninterested in science but want their taxes reduced.

On the other hand, science has also become very popular, so although it is opposed by some, it is supported by others. Today's curiosity-driven demand for science news pays for special magazines, newspaper columns, television and radio shows in many different countries and languages worldwide. Also, many people worldwide devote enormous personal efforts to obtain the advanced educations needed to pursue science careers.

It is possible that forces opposing science spending will attempt to defeat government research programs exploring Mars, including manned missions to Mars. However, I expect that although they might delay such missions for a few years, they will not be successful in permanently blocking them. Interest in mounting Mars missions has been separately expressed by the Russian Federation, by the European Space Agency, by the government of Japan, by the Peoples Republic of China, and by India, in addition to the USA. One or more of these will likely follow through in the next two or three decades.

World War II marked a turning point in public support of science. Applied science research by different countries during the war led to the nuclear fission bombs which abruptly ended the war with Japan, as well as to radar, jet engines for aircraft, electronic

computers, and long-range military rockets. Many in the public came to believe science should be supported by governments in order to enhance military security.

Since World War II most countries with modern economies have significantly subsidized domestic scientific research and technology development. The United States has been in the forefront of scientific research during that time, providing more funding than most other modern countries. It can be argued that this greater support of research and development is the primary reason that today the United States is the world's remaining superpower, with both the most capable military and the strongest economy. The direct benefits of basic scientific research, i.e., improvements in knowledge of the real world, are published in open journals available to all. However, enormous indirect benefits have accrued to the United States as highly educated, self-motivated people from countries with lower scientific funding levels moved here to pursue scientific and related technical work. As this dynamic has become increasingly understood, developing nations such as South Korea, China, and India have begun to expand their own government funding of scientific research and development.

Given that government scientific expenditures are in general positive, the question remains why some particular funding level should go to a Mars research program and how much is appropriate. **Mars research is especially worthy because it is likely to provide profound near-term advances in scientific understanding, with expected contributions to knowledge exceeding those from other scientific areas.**

During the past 60 years many research efforts have experimentally studied the fundamental properties of matter and energy using particle accelerators. Initially these studies produced much new knowledge. As particle energy levels increased over the years in order to study new regimes the cost of associated new accelerator facilities has increased with it. Productivity in terms of new knowledge obtained per unit of research effort invested appears to have declined substantially. Many elementary particles were newly discovered in the 1940s and 1950s but few new ones have been found recently. Such a decline may signal we have learned most of what can be practically studied in this field. Similarly in nuclear physics the Table of Isotopes has by now been filled in quite completely, with proposed additions having half-lives so fleetingly short that one wonders what the claim means that they exist. With declining research returns, some resources should be shifted from the study of fundamental matter and energy properties to more promising scientific areas.

The most productive areas of science today all seem to study complex ***arrangement patterns*** of matter and energy. For example, studies of the chemical and spatial interactions of atoms in large molecules, e.g., DNA or proteins, using both computer-based quantum mechanical modeling and experimental examination by instruments like the Scanning Tunneling Microscope (STM) which can image individual atoms and can even see individual electron orbitals, is rapidly expanding knowledge of molecular biology. New astronomical scans at wavelengths from the radio spectrum to infrared, visible and ultraviolet light, x-rays and even gamma rays are providing a wealth of

information ranging from the structures of our early universe soon after the "Big Bang" event about 14 billion years ago, to the supermassive black holes found at the center or our, and most other, galaxies, to the recently discovered extrasolar planets orbiting nearby stars. Ice cores drilled from Antarctica and Greenland have provided year-by-year samples of snow and trapped air allowing the reconstruction of climate and atmospheric variations for more than 200,000 years in the past. Computerized tomographic processing of accumulated seismic records has produced detailed 3D images of underground lava chambers such as the one under Yellowstone Park. Combined studies of the radiocarbon-14 and the oxygen isotope ratio contents of shells found on the ocean floor have produced a detailed record of inferred ocean sea level variations for the past half-million years, encompassing several ice ages. Statistical studies of mitochondrial DNA variations within presently living populations of various species, including humans, have reconstructed and dated past evolutionary events. Since such scientific studies of ***information patterns*** have recently been so productive, the desire for efficient use of resources would dictate increased funding in these areas at the expense of more mature but less productive fields.

The many material patterns existing on Mars provide an enormous new realm to study. The pressing astrobiology issue is whether alien life evolved there in parallel to Earth-life, and if not, then why not. Is evidence of alien life buried under Mars surface, even primitive pre-microbial fossils perhaps lacking any cell walls? Recent evidence that Mars had a magnetic field in the past begs the question how long its internal dynamo has been inactive. How does the size of Mars' molten core compare with Earth's? Does Mars have drifting crustal plates floating on an ocean of magma similar to Earth? Will Mars' volcanoes erupt again? What are Mars' surface and near-surface mineral deposits, where are they located and how did they form? How did Valles Marinaris, the longest and deepest canyon in the solar system, form? Were Mars' extensive networks of apparent river valleys formed by flowing water? Where did the water go? Do Mars' smooth northern plains hide a buried frozen ocean? How did Mars' climate evolve to its present state? What can be inferred from Mars' apparent global warming about Earth's apparent global warming?

Fundamentally, Mars investigations will study material structures existing in the martian crust and atmosphere on size scales ranging from atomic and molecular to planetary. It is likely that cores of ice and other layered deposits taken from Mars' polar caps could yield detailed information about the past similar to Earth's recently studied deep ice cores. Since Mars' crust appears to be layered as is Earth's, extensive study of regolith soil samples taken from different depths might eventually reveal much of Mars' history since formation of the solar system 4.5 billion years ago. Combining such information from Mars with similar data from Luna and from Earth may even unravel the 4.5 billion year history of our Sun.

3.2 Why isn't it sufficient to send tele-operated robots to Mars?

The arguments for relying only on robotic probes claim they are cheaper than sending human scientists and that they eliminate risks to human lives. Robotic probes have no life support requirements and can be abandoned after their mission is done. These arguments are correct, which is why robotic probes have been used extensively.

The counter to these arguments is that a solely robotic Mars exploration program would yield scientific insights vastly slower than a human presence can achieve. Furthermore, after a certain point when the "low hanging fruit" have all been picked by simple robotic probes, a human presence will be essential to continue the pace of progress. Robotic devices will continue to be used but they will be tele-operated by ***nearby*** humans.

It should be understood that robotic devices have limited capabilities. No robot has yet been built capable on its own of autonomously performing such a simple task as replacing a common household lighting fixture. At present, no autonomous robotic device can reliably walk on mechanical legs over ***irregular*** terrain, a basic capability mastered by most insects. The popular concept of robotic devices exhibiting human-like intelligence is simply a science-fiction fantasy since there has not yet been any success in creating machines which can think and understand. It is likely that future developments in creating intelligent machines will be paced less by Moore's Law of exponential reduction in computer memory prices than by the difficult neurobiological research into what is happening physically within human brains when we think or understand something. Researchers' apparent inability to make brain measurements which capture the complexity of mental activity seems to guarantee that the dream of intelligent robots will remain just that for the foreseeable future.

The robot probes sent to other planets are actually machines remotely tele-operated by humans. They do very little autonomously. They are highly engineered devices incorporating specialized structures designed to carry out a limited range of mission tasks. They incorporate computers programmed to store and retrieve command sequences, implement servo-control feedback loops, and avoid certain anticipated hazards. Although they have some flexibility, they are incredibly less flexible than a human and have no innate ability to respond to unanticipated developments.

An impediment to tele-operation by humans is the low bandwidth of round-trip information flow between robotic devices and their human operators. Low bandwidth imposes an enormous practical difference between remote control and being there. For earth-bound applications, this limit is set by available technology and its cost. If the recent rapid improvement of electronic technology continues, tele-operation of robotic devices on Earth will increase as the available round-trip bandwidth increases. However, even if future technology provides both very high bandwidth and extremely flexible robotic designs, human tele-operation of the robotic devices from a distance is fundamentally limited by the speed of light. Tele-operation of a machine (such as a remotely controlled car) by a human involves a feedback loop of sending commands,

monitoring the observed response to those commands, and then issuing corrective and additional commands. A round-trip time delay of 2.5 seconds such as would exist for a robotic car on Luna tele-operated from Earth could be accommodated by driving the car very slowly. However, the round-trip time delay between Earth and Mars can exceed 40 minutes depending on their relative orbital positions. For the two rover devices now operating on Mars this time delay has restricted their motions to only a few meters per day. Since human observations and responses occur in mere fractions of a second, a scientist located on Mars should be able to learn vastly more in a year than an earth-bound scientist tele-operating a robot on Mars. This is true even if the scientist on Mars is just tele-operating the same robot.

By going in person to locations under study on Earth, scientists have frequently been able to pose and answer successions of research questions within much shorter cycle times than the use of robots and photographs alone could achieve. On Earth, scientists find it effective to travel to remote and hostile places like the glaciers of Antarctica and Greenland. Scientists sometimes approach active volcanoes in person to install or retrieve measurement devices or to directly observe events. Although scientists have used robotic devices to help explore the ocean floor, they also study the ocean floor in person via manned submersibles. For example, the discovery in the late 1970s of giant tubeworm and clam species living at hydrothermal volcanic vent sites in the deep ocean was made in ALVIN, a Woods Hole Oceanographic Institute submersible which has typically carried two scientists and a pilot to depths reaching 4500 meters (14,764 feet) during its approximately 3700 dives.

There is no substitute for in-person exploration.

3.3 What parameters must characterize manned missions to Mars?

A sequence of manned missions to Mars will occur during the 21st century. Science will be the theme underlying all the missions. An analogous model is provided by the several scientific bases on Antarctica operated since the mid 1950s by the USA, Russia, and various other countries. Similar plans for an international scientific base on Luna, Earth's moon, are in development at present following the same analogy. Individual scientists would visit for months then return home but no individuals would live there permanently.

Unlike visits to Antarctica or even to Luna, travel to Mars would be necessarily more restricted. Orbital mechanics provides one-way travel window opportunities between Earth and Mars around opposition events, when Mars as seen from Earth lies opposite the sun. These recur roughly every 26 months when the two planets are aligned. Each mission would travel by rocket to Mars during one of these travel windows and return during the next. As dictated by orbital mechanics and the limitations of rockets, each mission would be away from Earth for 32 to 34 consecutive months of which 18 to 20 consecutive months would be spent on the Martian surface. Crews returning from Mars would pass in space their replacement crews traveling from Earth.

Mars mission crews would likely be multinational, consisting mostly of seasoned, highly trained research scientists but also including experts with specialized knowledge of engineering, mining, or medicine. Crews would be sufficiently large to locally cover all knowledge areas judged essential for emergencies, but they would also be augmented by larger expert groups participating virtually from locations on Earth, albeit with a difficult round-trip communications time delay.

All manned missions to Mars would be planned as round-trips, but the missions would leave Earth without rocket fuel for returning. Instead, rocket fuel for the return legs of each trip would be manufactured on Mars using martian resources, thus reducing travel costs from utterly prohibitive to affordable levels. Rocket fuel manufacturing would be accomplished by a nuclear-reactor-powered chemical factory remotely deployed in advance from Earth to a martian site located by accessible water-ice deposits. This factory would be tele-operated to produce fuel before the first manned mission to Mars leaves Earth. Subsequent missions would land at that same rocket refueling site which would become the de facto Mars base. Activities distant from that base would require long range surface mobility for which the nuclear-powered mobile engines of this thesis would be necessary. In addition, the nuclear engines would also power other equipment digging, trenching or drilling in the ground in order to obtain samples for scientific study, mine water-ice and other minerals, and construct pressurized underground rooms for fluids storage or for habitation.

See Appendix A for a detailed discussion of the parameters of manned Mars missions. These are dictated by unalterable constraints imposed by physical laws, by solar system characteristics, and by the limitations of existing materials, but are also to a significant extent influenced by the need for economy in an expensive undertaking. The important underlying facts are the following:
(1) The rocket fuel mass needed is an exponential function of the ratio of the speed change to be provided by the rocket divided by the rocket's exhaust speed.
(2) The orbital speeds of Earth (29.8 km/s) and Mars (24.1 km/s) are vastly greater than the exhaust speeds of any chemical rockets. Electric rockets do not have enough thrust for takeoff from a planet and nuclear thermal rockets have not yet been developed.
(3) Even the gravitational escape speeds of Earth (11.2 km/s) and Mars (5.0 km/s) are greater than the exhaust speeds of chemical rockets.

Included in this appendix are an explanation of the relevant aspects of rocket theory and technology, and a discussion of optimal Hohmann interplanetary transfers between Earth and Mars orbits along with a cost analysis of deviating from their restrictive transfer schedules. It concludes that existing rockets burning hydrogen with oxygen to develop an exhaust speed of 4.3 km/s already achieve nearly the highest exhaust speeds theoretically possible for chemical fuels, and the possible future development of nuclear thermal rockets which might significantly increase the exhaust speed is not assured. Only small deviations from optimal interplanetary transfers are possible without enormously increasing rocket costs through the exponential relation, so therefore only near-optimal

transfer schedules can be seriously considered. The characteristics of aeroentry, descent and soft-landing on Mars are next reviewed, since it would be utterly prohibitive to send sufficient rocket fuel from Earth to fully slow down a massive spacecraft arriving at Mars. The need to locally produce sufficient chemical rocket fuel for the return trip to Earth, using in situ resources on Mars, is also discussed and its proposed schemes are summarized. Martian production of the rocket fuel for returning is essential since the cost of sending from Earth enough fuel for the return home is also utterly prohibitive.

Logistical implications of these orbital transfer details and of the need for in situ martian resource utilization are summarized, including a tabulation of the maximum mass limits for cargoes such as nuclear engines that could be sent for different rocket options. NASA's expected capability will include soft-landing single payloads of up to about 100 tonnes on Mars' surface by using multiple launches of the new Ares5 rocket combined with their link-up in low Earth orbit. Single launches of the Ares5 will be able to soft-land on Mars payloads exceeding 20 tonnes.

Examples of some expected uses for the nuclear engines are also discussed. Finally, expected scientific activities consistent with mission goals and logistical constraints are examined. Each of these topics was considered in making design choices for the mobile nuclear engines.

Supporting orbital transfer computations implemented in MATLAB Notebook form and associated textbook citations for the numerical algorithms are found in Appendix B.

3.4 What problems need resolution for manned Mars missions?

There are many specialized design problems to solve in order to conduct manned missions to Mars. A human-sized centrifuge capable of providing artificial gravity during long space flights would help maintain health of the crew and could simplify hygiene, but none has ever been tested in space. Although it would be appropriate to use aerocapture at Mars in order to leave in orbit during the surface stay any deep-space components such as a centrifuge, a thermal radiator needed to control temperature in space, and a lightweight zero-gravity solar array, no spacecraft has yet used aerocapture to enter martian orbit. No previous rocket propulsion system design has protected its empty rocket fuel tanks during atmospheric reentry so that they could be reused. No nuclear-electric-powered chemical factory for rocket fuel production has ever been designed and fabricated, much less tele-robotically deployed on another planet. No water-ice mining equipment has ever been tele-operated on another planet. No pressurized buildings have ever been constructed on another body in the solar system, much less constructed from mostly local materials. Such design problems will require hard work to provide detailed engineering solutions, but paths to their solution are clear based on extensions of existing technologies.

The key difficulties which must be overcome are associated with providing significant engine power on Mars for both manned and tele-operated mobile vehicles and providing that power continuously during surface excursions over very long distances which last for long durations. Needed engine power levels range from less than 100 horsepower for small vehicles to several thousands of horsepower for digging and mining machines. Tonnes of martian soil must be excavated and moved to provide radiation shielding for the nuclear reactor deployed to power the local chemical fuel factory. Much more soil will periodically need to be excavated and processed in order to mine the frozen water it contains, then the harvested tonnes of water must be transported from the water-ice mining locations to the fuel factory. Much martian soil will need to be excavated and moved in order to construct additional pressurized spaces for the crew to use as scientific laboratories. Surface vehicles will also need to travel thousands of kilometers to collect regolith samples from many locations globally. At some of the locations, deep drilling equipment will need to be set up and powered in order to obtain crustal samples from different depths. Long distance vehicles will need significant cargo capacity in order to bring back their many samples from each excursion. For manned excursions, each engine must also power mobile life support systems which recycle water and breathing air while providing needed heat.

Although an operating fuel/oxygen factory on Mars could fuel muscular lightweight mobile engines such as spark-ignition internal combustion engines in the factory's immediate vicinity, it is not practical to transport sufficient fuel and oxygen to power long distance surface vehicles traveling thousands of kilometers from the fuel factory site. In addition, chemically powered engines cannot be used initially when needed during factory deployment since there would not yet be any accumulated free oxygen, hydrogen or methane for them to burn. In my opinion the mobile nuclear engine solution pursued herein is the only proposal that comprehensively addresses these key difficulties.

3.5 How do mobile nuclear engines help address those problems?

It is well known that very small assemblies of nearly pure fissile material can sustain a fission chain reaction and that vastly more energy is available from fission fuel than from the same mass of chemical fuel. Since the fission energy released is about one megawatt-day per gram of material fissioned, an assembly continuously producing heat at a one megawatt power level would only consume 0.55 kilograms of fuel through fission during an expedition's entire 18 month stay on the martian surface. If an engine thermally converted that stream of heat into work with a thermal efficiency of 25%, the engine would continuously develop 335 horsepower. That engine power level is well within the range needed for mobile martian vehicles in order to travel across long distances and excavate soil. These facts show that if mobile engines based on fission can be practical on Mars at such power levels, the high energy content of their fuel automatically confers an ability to travel globally over essentially unlimited distances and durations, thus providing researchers with significant engine power anywhere on Mars' surface for mobility or for the operation of auxiliary equipment.

CHAPTER 4
4 NUCLEAR CONFIGURATION CHOICES

All engineering designs consist of making a set of choices. Some of these choices involve the selection of one option (e.g., a material) from lists of admissible options. These configuration choices are the main focus of the present chapter. Other design choices concerning selection of numerical parameters (e.g., the size of components) are addressed later.

4.1 Goal: Minimize Mass For Specified Output Shaft Power

An important backdrop for configuration choices is how the choices are made. Criteria for design choices are identified here.

Nuclear powered engines were developed in the past for applications on Earth for which they contributed a competitive advantage over existing conventional engines powered by chemical combustion. Some competitive advantage was necessary since nuclear engine designs were all far more massive and expensive than comparable engines powered by combustion. Nuclear powered jet engines were developed in the late 1940s and early 1950s because the high energy content of their fissile fuel would confer strategic bomber aircraft with far greater intercontinental roundtrip ranges than previous aircraft. However, nuclear powered aircraft were never deployed because mid-air refueling, which greatly extended range of combustion powered aircraft, became established practice in the mid 1950s and because far faster intercontinental missiles using suborbital rockets soon thereafter made strategic bombers obsolete. Nuclear powered engines for naval submarines were developed in the same time period because their ability to operate without consuming free oxygen would extend maximum durations of deeply submerged missions from days to years. Subsequently, the new nuclear naval propulsion engines were also used to propel large surface ships (e.g., aircraft carriers) in order to reduce their needs for tanker ship refueling convoys.

Combustion engine technology is not a serious competitor to nuclear powered engines in locations like Mars lacking either combustible fuel or free oxygen. However, nuclear engines for Mars should be expected to share known characteristics of previous nuclear designs. In particular, nuclear powered engines for Mars surface vehicles are expected to be far more massive than terrestrial combustion engines of the same output power ratings.

Many different cost functions could reasonably be used to optimize different aspects of previous Earth-based nuclear designs. However, the Mars situation is different because engine fabrication costs are trivial in comparison with Mars delivery costs. The extremely high cost per unit mass to send any payload from Earth to Mars' surface leads to adoption of the Nuclear Engine Mass Delivered from Earth as the single most important cost function to be minimized. In addition, values of Nuclear Engine Mass

Delivered from Earth exceeding a certain high threshold value would be too massive for delivery by any rocket vehicle expected to be available during the present century.

Some nuclear engine designs would substitute martian surface materials to reduce the shielding mass sent from Earth. Even though their delivery cost would be reduced, excessive total vehicle mass would still be a result to be avoided in the interest of vehicle mobility. Therefore, mass weighted by appropriately chosen factors could still provide a useful cost function to minimize, even for those designs. **It follows that the minimization of a cost-weighted vehicle mass is the main tradeoff criterion used to make design optimizing choices.**

4.1.1 Mass Savings As Forecast By An Approximate Shielding Model

The following approximate model of shielding mass savings is a theme motivating many design choices. The main point of this model is that although the necessary *thickness* of a radiation shield does not depend on the physical size of the radiation source which it attenuates, the necessary *mass* of the radiation shield does depend sensitively on the source's physical size. Reduction of radiation source size can greatly reduce the necessary shielding mass.

We consider q_s, a spherical radiation source of radius r_s, surrounded by a radiation shield of uniform mass density ρ configured as a spherical shell of thickness τ. Neglecting non-radial propagation and initially ignoring the shield's attenuation, the radiation intensity at location r, where $r>r_s$, would be

$$I_{\text{bare}} = \frac{q_s}{4\pi r^2} \tag{4-1}$$

Including the shield, if we ignore scattering and model the shield as simply absorbing radiation without re-emission with a macroscopic cross-section of σ , the radiation intensity at location r, where $r>r_s+\tau$, becomes

$$I_{\text{shielded}} = \frac{q_s}{4\pi r^2}\exp(-\sigma\tau) \tag{4-2}$$

Thus, the required shield thickness is directly determined by the attenuation factor needed and is independent of the physical size of the radiation source:

$$\tau = \frac{1}{\sigma}\ln\left(\frac{I_{\text{bare}}}{I_{\text{shielded}}}\right) \tag{4-3}$$

However, the shield's mass follows from its assumed spherical shell geometric shape and depends strongly on the physical size of its enclosed radiation source.

$$M_{\text{shield}} = \frac{4}{3}\pi\left((r_s+\tau)^3 - r_s^3\right)\rho = \frac{4\pi\rho\tau^3}{3}\left(1+3\left(\frac{r_s}{\tau}\right)+3\left(\frac{r_s}{\tau}\right)^2\right) \tag{4-4}$$

Since the minimum possible shield mass would result if the radiation source were vanishingly small, this result can be written more compactly as the ratio of the actual

shield mass needed for a spherical source to the shield mass that would be needed for a point radiation source of the same intensity.

$$\frac{M_{\text{Actual Shield}}}{M_{\text{Ideal Shield}}} = \left(1 + 3\left(\frac{r_s}{\tau}\right) + 3\left(\frac{r_s}{\tau}\right)^2 \right) \qquad (4\text{-}5)$$

Some selected values of this function are listed in Table 4-1.

Thus, for a given rated reactor power level we might estimate a necessary radiation shield thickness and mass assuming that the reactor behaves as a point source of radiation. If the actual reactor were spherical with a radius of 10% of the estimated necessary shield thickness, the shield mass would need to be increased by 33%. If the reactor's radius were 26.4% of the estimated necessary shield thickness, the shield mass would need to be doubled. If the reactor's radius were equal to the necessary shield thickness, the shield mass would need to be multiplied by 7. If the reactor's radius were 5 times the necessary shield thickness, the shield mass would need to be multiplied by 91.

In truth, actual shield behavior is more complicated than this simple model. An actual shield design must contend with non-radial propagation of radiation, with scattering and re-emission at different energies and directions, and even with the generation of gamma ray photons within the shield itself by inelastic neutron scattering and by neutron absorption. However, to whatever extent this simple model approximates reality, its impressively large mass penalty factors hint that reductions in reactor size may remarkably reduce reactor shielding mass requirements. Consequently, priority is given herein to any features with the possibility of reducing reactor size.

4.2 Output Shaft Power Design Ratings

Members of the family of nuclear engine designs developed herein differ in their rated values of maximum continuous output shaft power. The engine family's entire range of output shaft power ratings varies from 100 horsepower to 10,000 horsepower in order to cover anticipated needs discussed in Chapter 3. The range is represented herein by nine different output shaft power ratings distributed logarithmically as listed in Table 4-2.

4.3 Nuclear Engine Thermal Conversion Overall Configuration

The overall configuration chosen for these mobile nuclear engines is the Nuclear Heated Open Brayton Cycle diagramed in Figure 4-1.

4.3.1 Open Brayton Cycle Description

The Open Brayton Cycle provides an effective implementation of mobile engines on Mars. As depicted in Figure 4-1, low pressure martian air is first compressed to an elevated pressure, then heat is added at that constant elevated pressure, then finally the

Table 4-1: Mass Penalty Factor for Shielding A Spherical Radiation Source

$\left(\frac{r_s}{\tau}\right)$	$\frac{M_{\text{Actual Shield}}}{M_{\text{Ideal Shield}}}$
0	**1**
0.1	**1.33**
0.26376	**2**
0.45743	**3**
0.61803	**4**
0.75831	**5**
0.88444	**6**
1	**7**
2	**19**
3	**37**
4	**61**
5	**91**

Table 4-2: Output Shaft Power Design Ratings of Nuclear Engines For Mars

Engine Model Number	**Output Shaft Power Design Rating**	
	horsepower	**kilowatts**
1	**100**	**74.6**
2	**178**	**132.7**
3	**316**	**235.9**
4	**562**	**419.5**
5	**1,000**	**746.0**
6	**1,780**	**1,326.6**
7	**3,160**	**2,359.0**
8	**5,620**	**4,195.1**
9	**10,000**	**7,460.0**

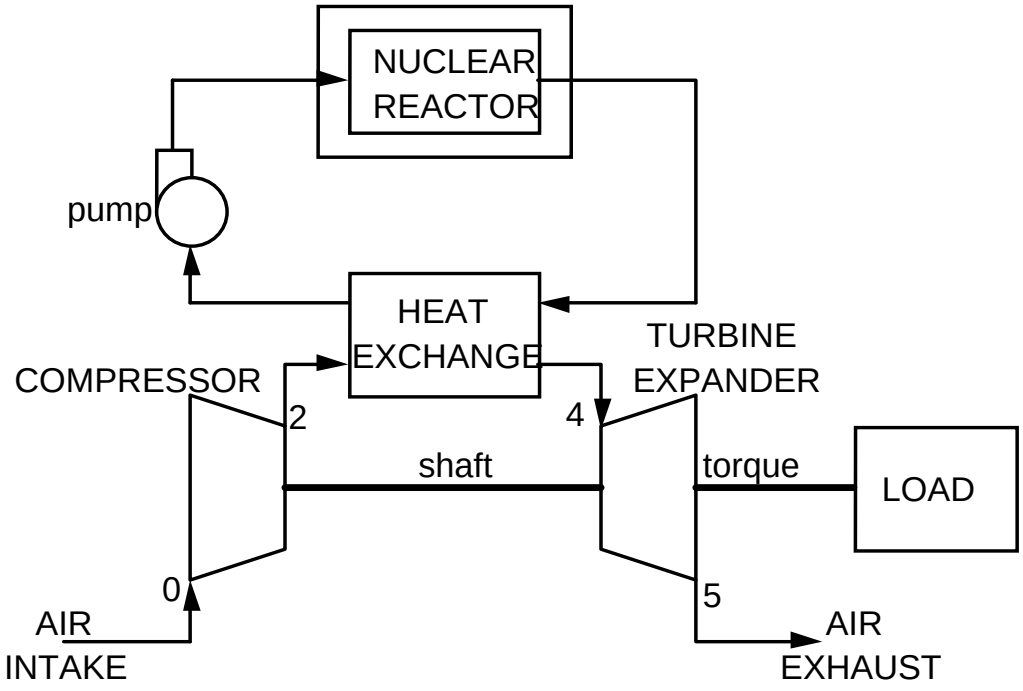


Figure 4-1: Nuclear Heated Open Brayton Cycle

compressed, heated air expands back to the lower martian air pressure through a turbine expander mounted on the same drive shaft as the compressor. Expander work exceeds compressor work and provides net power to an external load in the form of shaft rotation torque. If needed, the shaft output power can be converted to electrical, pneumatic, or hydraulic forms to be used by various different types of powered equipment.

The advantage of adopting an open cycle thermal conversion system is that neither a true radiator nor a low pressure heat exchanger is required to reject the substantial flow of waste heat. Instead, waste heat is rejected effortlessly as the enthalpy difference between the cold air intake and the hot air exhaust airflows, shown as locations 0 and 5 respectively in Figure 4-1.

4.3.2 The Importance of High Temperature

Since gas temperature is raised by a compressor, the compression pressure ratios achievable in Brayton conversion cycles are fundamentally limited by the temperature range. In general, low compression ratios in Brayton cycles impose an even more severe thermal conversion efficiency reduction penalty than is imposed on ideal Carnot cycles by high temperature limitations. For Open Brayton Cycle implementations using the low pressure martian air, the efficiency penalty paid for limiting high temperature is even more severe since the lowest pressure stages of the compressor and the turbine expander will unavoidably operate with somewhat compromised component efficiencies.

Good thermal conversion efficiency is important for mobile nuclear engines on Mars because poor thermal conversion efficiency would require increasing the reactor's thermal power to maintain the same output shaft power at specified design ratings. That increased fission power would increase source radiation which in turn would require more massive radiation shielding to limit leakage to the same radiation levels. Thus, total nuclear engine mass, not fissile fuel conservation, is the primary reason that it is essential to pursue a high temperature reactor design. It is also important to provide some additional margin for temperature drop beyond that needed for thermal conversion since heat exchanger component mass can be traded off against some irreversible temperature drop. For all of these reasons, the configuration choice of Figure 4-1 implies that **it is very important for engine performance that the nuclear reactor must be able to heat compressed martian air to the highest feasible temperature.**

4.3.3 Avoidance of Direct Heating of Martian Air Within Reactor

A far simpler alternative implementation of the Nuclear Heated Open Brayton Cycle could have been chosen by entirely eliminating Figure 4-1's reactor coolant loop along with its coolant circulating pump and its coolant-to-air heat exchanger. Instead, the martian air emerging from the compressor would itself have been routed directly through the reactor then to the turbine expander. In addition to being simpler, this alternative implementation would also have achieved higher air temperatures with identical reactor temperatures, thus increasing thermal conversion efficiency.

This alternative implementation was rejected for two reasons. First, based on heat transfer considerations alone, the use of a gas to absorb reactor heat instead of a far higher density liquid would require more heat transfer surface area and consequently, a larger fission reactor volume. That larger fission volume source of neutron and gamma radiation would in turn necessitate more massive radiation shielding in order to limit crew member radiation doses to identical levels.

Second, oxygen atoms in CO_2, nitrogen atoms and argon atoms, all of which are present in the martian air, would become activated during their passage through the reactor. Activation products would include O^{19} which emits gamma ray photons of 0.198 MeV and 1.357 MeV as it beta decays with a 26.8 second half-life, N^{16} which emits gamma ray photons of 6.13 MeV and 7.12 MeV as it beta decays with a 7.1 second half-life, and Ar^{41} which emits a 1.3 MeV gamma ray photon as it beta decays with a 1.83 hour half-life. Some O^{18} would also be converted by a (n,d) reaction to N^{17} which emits delayed neutrons with a 4.17 second half-life then adds to the inventory of decaying N^{16}. Thus, direct routing of the compressed martian air through the reactor would result in the engine's air exhaust plume becoming an intense emitter of gamma and neutron radiations. Some massive shielding would be needed to protect crew members from the exhaust plume when they are near the operating engine. Thus, the Figure 4-1 configuration was chosen over the simpler alternative to reduce total engine mass.

4.4 Choice of Reactor Coolant: Molten Lithium

A reactor coolant fluid can be considered good for this application if a low power pumping system circulating it at a modest flow rate can transport at high temperature a large thermal power with only small heat transfer film temperature drops. It is also important that it does not create other problems such as additional radiation or fluid leakage or fluid decomposition or boundary corrosion.

Among the liquids available for use as reactor coolants, the best heat transfer performance with good flow properties is obtained from alkali liquid metals. Since each is an element, none of the molten alkali metals decompose at high temperature as do some molecular liquids. Of the alkali metals, molten sodium (Na) has been used most extensively for nuclear applications and has the largest experience base since Liquid Metal Fast Breeder Reactors (LMFBRs) employ it as their primary coolant fluid. The experience base shows sodium produces far less corrosion in LMFBR systems than is typical in Pressurized Water Reactor (PWR) systems; this result is expected to apply also if other alkali metals are used as reactor coolants provided that chemically compatible contacting materials that do not dissolve [Borgstedt and Guminski 2000] are used. Regardless of sodium's larger experience base, molten lithium (Li) has two significant advantages over sodium when applied to mobile nuclear engines on Mars.

First, molten lithium's boiling point temperature is higher than the boiling point temperature of molten sodium (or of any other alkali metal) at any equal reactor vessel pressures. Since reactor vessels' internal pressures must be limited to avoid overstressing their wall materials, this implies lithium confined in a reactor vessel might achieve the highest possible operating temperature. The normal boiling point temperature of lithium at Earth's sea level atmospheric pressure is 1620 Kelvins (i.e., 1347C), whereas the normal boiling point of sodium is only 1156 Kelvins (i.e., 883C). Figure 4-2 (extracted from Appendix C) plots lithium's saturation pressure vs. temperature, showing both published experimental data points up to 2000 Kelvins and my own cubic spline fit to the *logarithm* of those data points. My extrapolated estimate of lithium's boiling point temperature at the clearly practical 15.5 MPa (2250 psi) operating pressure of PWR reactor vessels is 2529 Kelvins (i.e., 2256C). Since lithium's heat transfer properties are similar to other alkali metals, its higher temperature performance recommends it for high temperature reactor designs.

Second, a disadvantage of using molten sodium as a reactor coolant is that neutron capture in the sodium forms the isotope, Na^{24}, which decays with a 15 hour half-life to stable magnesium while emitting two gamma ray photons of fairly high energy, 1.38 MeV and 2.75 MeV, as part of its beta decay process. Thus, if molten sodium were the reactor coolant, the entire coolant loop and in particular the coolant-to-air heat exchanger in Figure 4-1 would need to be surrounded by massive gamma ray shielding. **In distinction to sodium, lithium does not produce any gamma ray emitters as a result of activation by neutrons.** If molten lithium were the reactor coolant, the coolant-to-air heat exchanger in Figure 4-1 would not need radiation shielding.

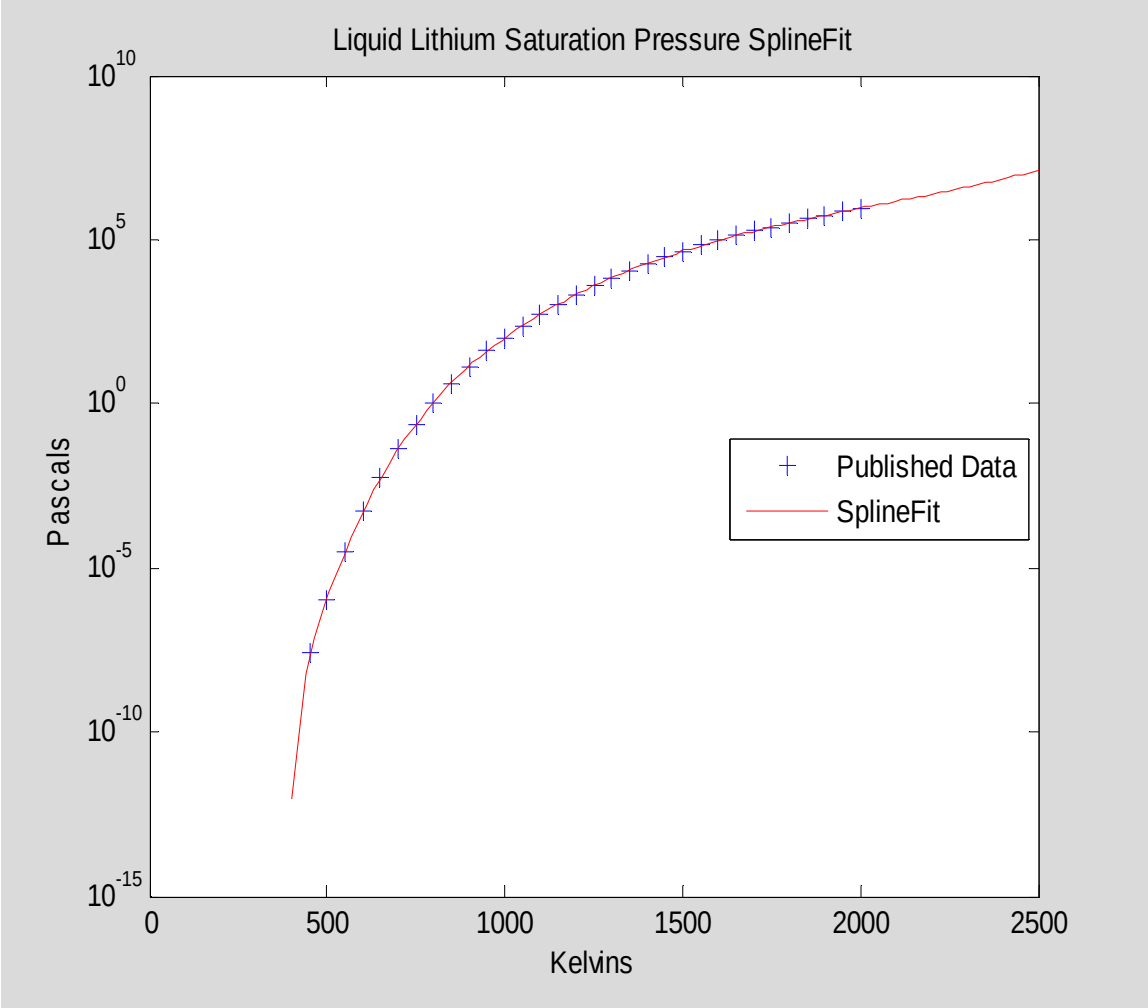


Figure 4-2: Molten Lithium Boiling Point Temperature Vs. Pressure

Lithium occurs as a mixture of the two isotopes, Li^6 and Li^7. For a thermal spectrum reactor the high thermal neutron absorption cross section of Li^6 (i.e., 940 barns) makes it unsuitable for use as a coolant since it would poison the fission chain reaction, but isotopically pure Li^7, having a thermal neutron absorption cross section of only 0.033 barns, could serve well in a thermal spectrum reactor. For a fast spectrum reactor, the naturally occurring isotope mixture could be used as the reactor coolant. The elastic scattering cross sections of both lithium isotopes are small, so in spite of lithium's low atomic weight it is not a good neutron moderator.

For terrestrial nuclear reactors, lithium has not yet offered any compelling competitive advantages over more conventional coolants. Commercial applications have not valued either the higher thermal conversion efficiency which lithium's higher temperature capability makes possible or the reduction in radiation shielding which its low activation nature permits. Uranium fuel and radiation shielding are each cheap, so lower temperature has been valued since it is perceived as conferring higher reliability.

Although lithium-cooled reactors have not been deployed, their potential superiority in certain niche uses has long been recognized. The original 1967 edition of Nuclear Reactor Engineering [Glasstone and Sesonske 1967, 713] saw lithium-cooling as a promising "future possibility" due to its low melting point and high boiling point temperatures, its high specific heat and its high thermal conductivity. In the 1980s funded SP100 project to develop 100 kw(e) nuclear reactor electric power sources for use in space, molten lithium was adopted as the reactor coolant in all design versions. More recently, Project Prometheus adopted molten lithium as the reactor coolant chosen for its nuclear reactor powered electric ion rocket to be designed to propel the Icy Moons Orbiter spacecraft to closely study Jupiter's four Gallilean moons. The DOE is also considering using lithium as the reactor coolant in its Very High Temperature Reactor (VHTR) designs being developed for hydrogen production applications.

4.4.1 Magnetized Coolant Pumping by DC Current Injection

Since the chosen molten lithium coolant has high electrical conductivity, it is feasible to use a pumping scheme without mechanical moving parts. This is attractive because it enhances reliability. The pumping configuration is depicted in the Figure 4-3 diagram. Here, a low voltage electrical power source drives a direct electrical current, **I**, through electrodes galvanically connected to two opposing walls of a rectangular cross section lithium flow duct, forcing much of that electrical current to flow through the molten lithium filling the duct. An external magnet not depicted in Figure 4-3, either a permanent magnet or an electromagnet, applies a steady magnetic field, **B**, to the same current-carrying molten lithium but in a direction oriented perpendicular to the other two opposing duct walls. The resulting electromagnetic force on the lithium pushes it in the flow velocity direction labeled **V**. The pumping pressure head provided by this scheme is proportional to the product of the electrical current and the magnetic field.

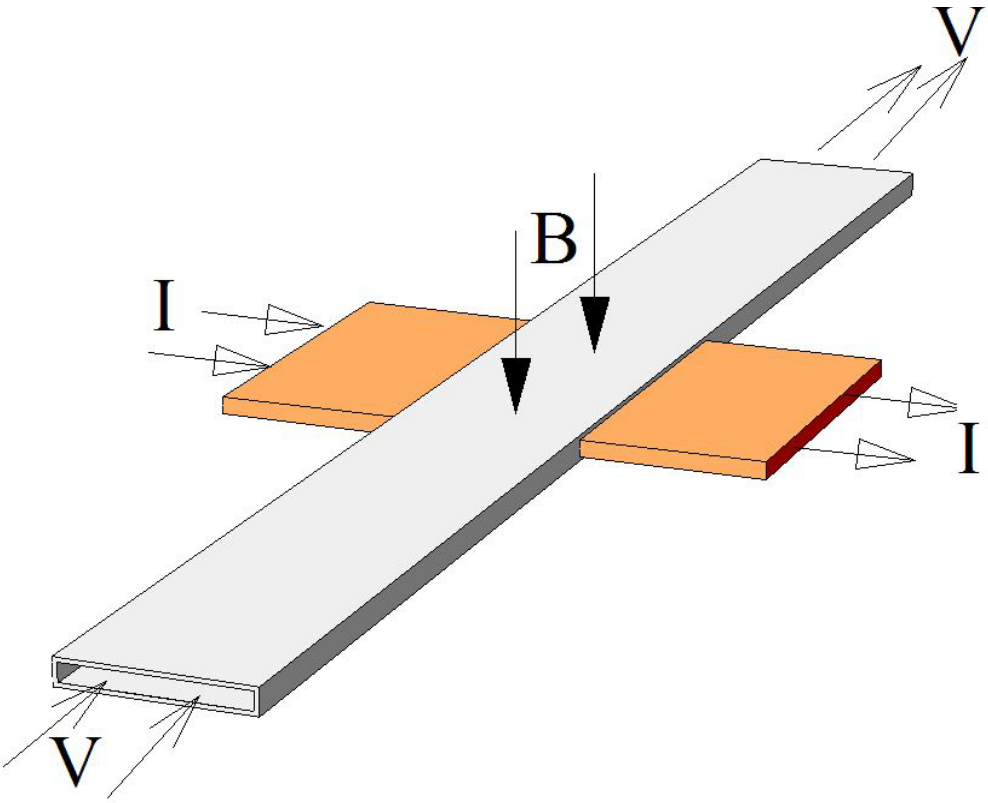


Figure 4-3: Molten Lithium Pumping By Injecting DC Electrical Current

There are multiple alternative technologies for pumping liquid metals. The chosen DC current injection method is attractive since it is simple and has no mechanical moving parts. A recent survey [Polzin 2007] of liquid metal pumping technologies examined and compared with other technologies six available off-the-shelf single-stage DC liquid metal pumps having rated flows ranging from 10 gpm to 8,300 gpm and rated pressure differences ranging from 138 kPa to 517 kPa. Their efficiencies ranged from 12% to 50%. The particular DC pump working with lithium at 1423 K provided a single-stage pressure difference of 138 kPa at a flow rate of 420 gpm with an efficiency of 19%.

4.4.2 Choice of Reactor Coolant Pressurizer Gas: Argon

Any liquid coolant confined within a fixed volume pressure boundary must be provided with some way to accommodate the changes in coolant volume resulting from temperature changes. This nuclear engine design follows conventional practice in providing pressurized gas in direct contact with the liquid coolant in a portion of the fixed volume known as the *pressurizer*. In this way, changes in the liquid coolant volume are matched by opposing changes in gas volume, thus limiting peak pressures.

Because molten lithium is very active chemically, many gases cannot be considered for pressurizer use. Any of the noble gases could in principle be used since none of them chemically reacts with lithium. However, argon is chosen here because it is far less

likely to be lost through tiny leaks than helium and because if needed it could be replenished on Mars. Indeed, argon constitutes 1.6% of the martian atmosphere and could be separated from the martian air by a small electrically operated cryogenic system.

4.5 Choice of Solid Materials Contacting Reactor Coolant: W, Ta, Mo

Although lithium has been of continuing interest to controlled thermonuclear fusion research communities for its possible uses in a fusion reactor blanket and to other groups for other uses, it chemically attacks many materials, especially at high temperature. Hanford Engineering Laboratories' 1978 internal technical report document TC-1000, Lithium Literature Review: Lithium's Properties and Interactions [Jeppson et al. 1978], provides tables listing lithium's long-term chemical compatibility with many materials for different operating temperatures. Metals such as aluminum, magnesium, tin and zinc are attacked at all temperatures. Low-carbon and low-chromium steels must be restricted to temperatures below 400C to avoid attack. Austenitic Cr-Ni stainless steel can operate up to 500C in contact with lithium. Ferritic-chromium stainless steel can operate up to 800C in contact with lithium. However, the tables listed four other metals, i.e., niobium (a.k.a. columbium), tantalum, molybdenum, and tungsten, as being compatible with lithium up to at least their maximum listed temperature, which was 900C.

After more than two decades of further investigation by the space nuclear power community and the fusion community, the high temperature lithium compatibility situation was summarized in the Journal, Fusion Engineering and Design in a paper [Zinkle and Ghoniem 2000] containing the following statement:

> **"The Li chemical compatibility data base for refractory alloys can be summarized as follows: the alloy T-111 (Ta-8W-2Hf) has good compatibility with lithium for exposure temperatures up to 1370C (both static and circulating loop experiments). Similarly, the existing corrosion data for Nb-1Zr exposed to lithium indicates good compatibility up to 1000C (static and circulating loops). Pure tungsten and W alloys are generally compatible with lithium up to 1370C (attack observed at 1540C). Mo alloys (TZM) also have good compatibility with lithium up to 1370C (attack observed at 1540C). "**

Thus, niobium is not quite as good for use with lithium as the other three refractory metals. Other materials assessed in that publication included ferritic-martensitic (F/M) steel and silicon carbide (SiC), both of which must be limited to 600C when contacting lithium, and vanadium (V) alloys, which can operate with lithium up to 700C.

Additional information was provided in the October 2002 document by the same two co-authors and others, EVOLVE Final Report, which is posted on the UCLA Fusion Studies Group's web site, http://www.fusion.ucla.edu/. This report's name is an acronym for a high temperature molten lithium blanket system to use in future fusion reactors. It

contains information on materials relevant to the present nuclear engine design. Its lithium chemical compatibility information includes the following statement:

> **"The W/Li compatibility limits of 1370C… are based on an extensive series of tests by J. A. DeMastry (Batelle-Columbus Labs) and other researchers performed in the mid-1960s. DeMastry observed good compatibility during 1000 hour isothermal capsule tests at 1370C, but corrosion/dissolution during exposures at 1540C and 1650C. Since dissolved tungsten was detected during post-exposure chemical analysis of the lithium, it seems unlikely that tungsten would be compatible with lithium for long-term (>>1000 hour) operation at temperatures above 1400C. On the other hand, some lithium heat pipe tests have found that W/Li was satisfactory for medium term operation up to 1600C. Mixed results on W/Li corrosion have been obtained in several other studies at temperatures >1400C. "**

The EVOLVE report proposes use of tungsten or its alloys and it reports on the fabricability of complicated assemblies using candidate tungsten alloys. The report seems to find the most promise in tungsten alloyed with rhenium, i.e., W-5%Re.

Based on all this information, it appears appropriate to limit the maximum nuclear engine design temperature for molten lithium to not exceed 1643 Kelvins (i.e., 1370C) and to use any combination of tantalum, molybdenum, tungsten or their lithium-resistant alloys with other metals as the solid surface materials contacting molten lithium. Within this small set of admissible materials, tungsten or one of its alloys is preferred because of tungsten's other good properties concerning radiation shielding, heat conduction, mechanical strength, and melting temperature (i.e., 3695 Kelvins or equivalently 3422C is the highest melting point of all elements). Tungsten's only apparent defect is that it is difficult to machine or weld.

Thus, the main solid material chosen for contact with molten lithium is tungsten, but molybdenum and tantalum will also be used if needed.

4.6 Fuel Cycle Configuration Strategy: Replace Entire Core to Refuel

It is expected that the largest contributions to the total masses of optimized engine designs will be from radiation shielding and thermal conversion components, and that the fuel in the reactor will be only a tiny fraction of total engine mass. **Since each engine will be used by multiple consecutive expedition crews, the total engine mass deployed from Earth to Mars over decades of visits will be minimized by designing the engines to be refueled by crews on Mars.** This need for refueling reactors on Mars is a significant requirement not easy to implement. However, it is worthwhile due to its enormous reduction in the total mass sent over time.

The chosen fuel cycle configuration strategy for these engine designs treats the entire reactor core, including all of its fissile fuel, as a single replaceable module. Spare reactor core modules sent from Earth will be far smaller and less massive than an entirely new engine with all of its radiation shielding. However, because of this strategy, there must be provisions for opening the reactor, for removing its spent core and replacing it with a spare, then reclosing and resealing the reactor. Refueling will be accomplished through robotic devices tele-operated by nearby crew members on Mars' surface, but the nuclear engine designs must include features addressing the following issues:

(1) Spent fuel cooling must be provided during the refueling process to accommodate ongoing fission product decay heat production,

(2) Molten lithium must be prevented from freezing (at about 182C) during the refueling process, since lithium solidification might prevent either removal of the spent core or insertion of the new core,

(3) Molten lithium should be prevented from contacting martian air, even during refueling. At Earth's sea level air pressure, molten lithium burns violently with a flame in either pure CO_2 or pure N_2 gases, each a major component of the martian air. It is unknown whether lithium would burn in the low pressure martian air, but it seems likely there would be a chemical reaction. Of course, argon gas, also a major component of the martian air, is suitable for contact with molten lithium if purified.

After refueling the engine, no effort will be made to recycle the unfissioned fuel remaining in the spent fuel core. Instead, each spent core will be placed in its own sealed container with neutron absorber material. Filled spent fuel core containers will be placed in a "boneyard" disposal site on the martian surface not far from the Mars base where successive crews from Earth will land and depart. Multiple spent fuel core containers will be separated from each other in that boneyard by sufficient distance to make criticality accidents incredible. External surface areas of the spent fuel core containers, extended by fins if needed, will be sufficient to guarantee that natural convection of the martian air augmented by radiatiation to the sky will passively remove all decay heat without exceeding melting point temperatures of the container materials. A labyrinth surrounding the boneyard will also be constructed by piling up martian soil into 3 meter tall berms. It will protect people at nearby martian surface locations from line-of-sight exposure to the decay gamma rays emitted by the spent fuel cores while still allowing surface access through the labyrinth to the boneyard by tele-operated robots. The disposal site will also be electronically monitored.

4.7 Choice of Reactor Fuel Shapes: Fuel Plates

The fuel shape with the largest experience base is the cylindrical fuel pin or fuel rod. Most stationary reactors use pin bundles with pin diameters of 5 to 10 mm, including cladding. This shape can provide a large heat transfer surface area per unit fuel volume. Reactor coolant flows axially between pins. The second most common fuel shape is the spherical pellet. Pellet grains fill a reactor chamber which supports them collectively and a coolant gas flows in the spaces between them.

The "fuel plate" shape of Figure 4-4 chosen here for use in nuclear engines for Mars does not have a large experience base. It is not even mentioned in some nuclear engineering texts, e.g. , [Stacey 2001] or [Duderstadt and Hamilton 1976] but it is described in *Introduction to nuclear engineering* [Lamarsh and Baratta 2001, 310] as being appropriate for a ***mobile*** reactor (which has shock and vibration loads). The Figure 4-4 diagram shows a fuel plate section in which a finite slab of fissile fuel material is surrounded by cladding to form a sandwich structure. Fuel plates are typically thin (e.g., 1.5 to 5.0 millimeters) but extend much farther in their other two dimensions. A reactor core contains many such fuel plates arranged parallel to one another, forming a stack with small spaces between the plates through which reactor coolant flows. Robust mechanical support for the fuel plates, appropriate for the jarring vibrations of a mobile vehicle driving over rugged terrain, is provided by a surrounding rectangular enclosure in which grooves hold the individual plates in position by their edges. The conventionally configured stack of fuel plates cools them in parallel by having reactor coolant flow in the same direction in the parallel spaces between plates. However, an alternate coolant flow configuration can also be considered. In a way not possible with fuel pins or pellets, staggered fuel plates can simultaneously serve as flow baffles, as depicted in Figure 4-5

Here, grooves hold each fuel plates on three edges, the alternately left or right side as shown and the perpendicular edges. Coolant flowing upwards through this reactor core configuration is forced by the fuel plate baffles to flow alternately to the left and right as it ascends. This can yield a low coolant flow-rate high ΔT design not feasible with either pins or pellets, so that in spite of the low coolant flow rate a high thermal power can still be transported from the reactor to the compressed air heat exchanger of Figure 4-1.

4.8 Reactor Fuel Material: Highly Enriched Uranium Nitride

In order to minimize the physical size of the reactor core and thus minimize the necessary shielding mass, the fissile nuclide chosen should in principle have the largest possible fission cross-section and its atom density should also be as large as possible. However, there are also other competing considerations. **The highly enriched UN fuel choice adopted here is a compromise between these considerations.** It does not yield the smallest possible reactor core but its core is small and it avoids other difficulties.

Minimum reactor core size in a design is set either by thermal hydraulic heat transfer considerations or by limitations on criticality. Here it is assumed that because of the excellent high temperature characteristics of molten lithium, criticality issues rather than heat transfer issues will determine minimum reactor core size. It then immediately follows from comparing fission cross sections that plutonium-239 would be a better fissile nuclide for this use than uranium 235, and also that pure metallic Pu^{239} fuel could have a much smaller critical configuration than could either any chemical compounds of plutonium or uranium or any of their metallic alloys with nonfissile species.

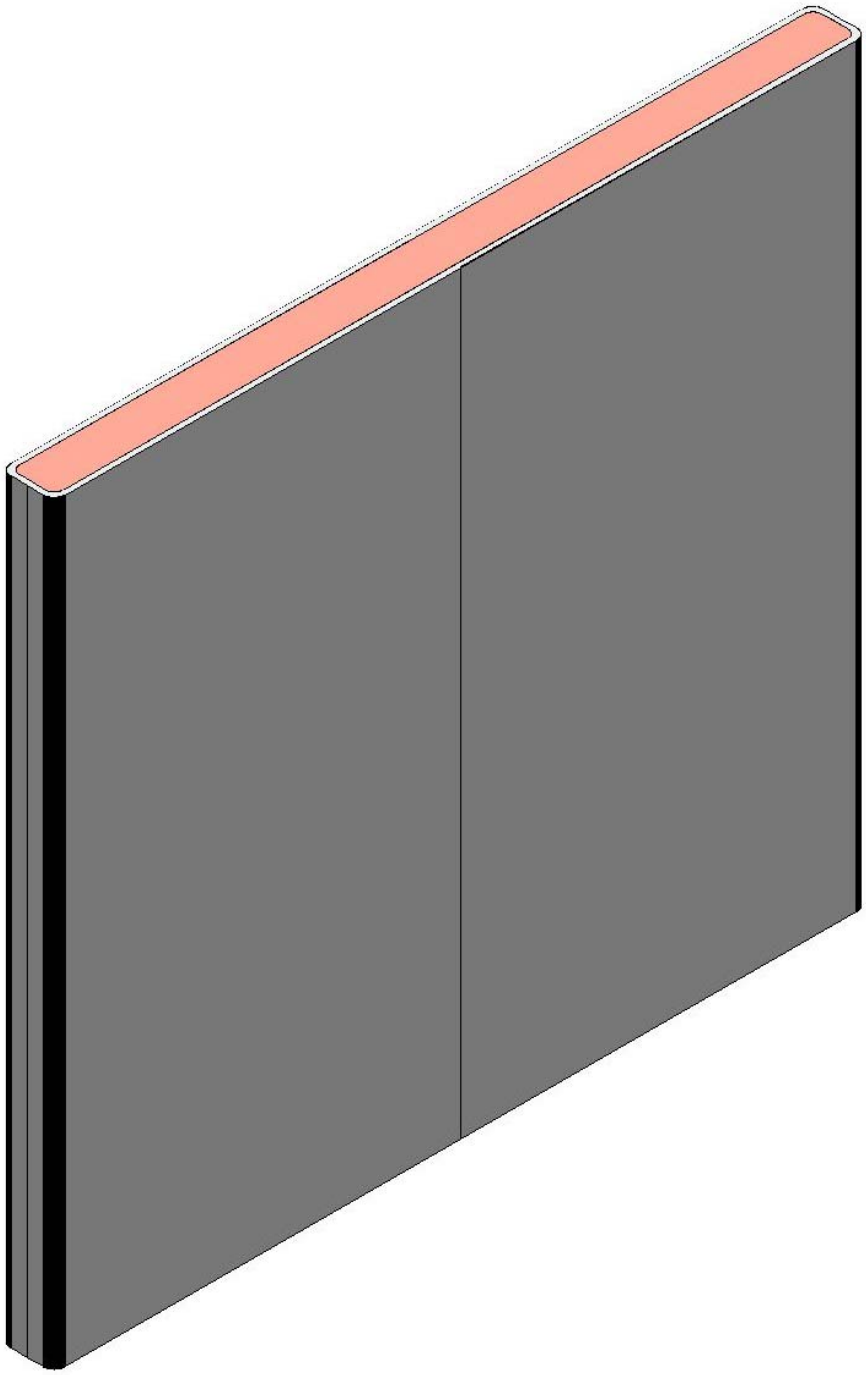


Figure 4-4: Section Cut Through Fuel Plate

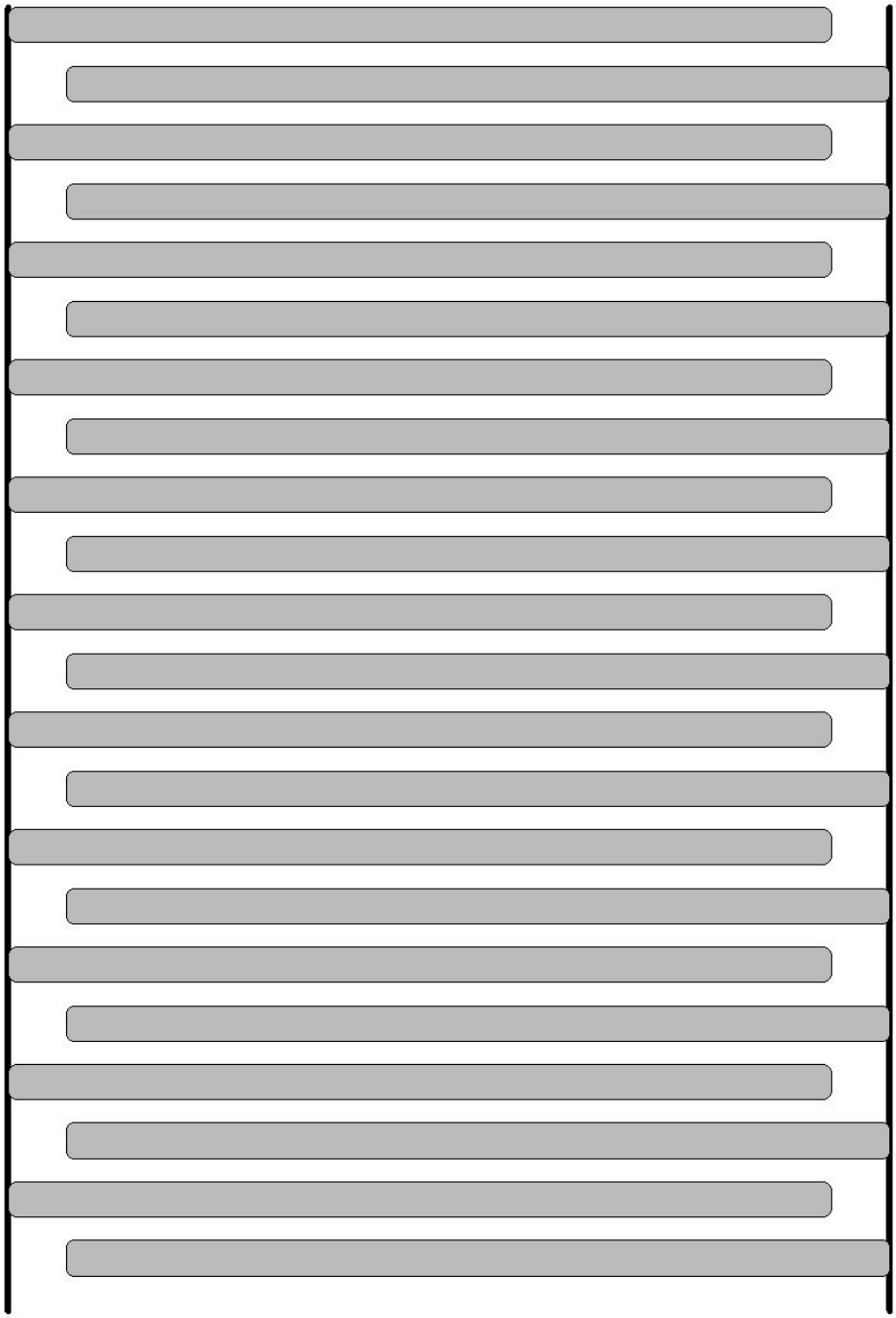


Figure 4-5: Baffled Reactor Coolant Flow Via Staggered Fuel Plates.

However, in spite of the possibility that an innovative design using Pu^{239} fuel could yield the lowest possible nuclear engine mass, plutonium fuel was rejected for this thesis work based on several concerns. First, if a rocket launching a nuclear engine cargo from Earth were to fail, any plutonium fuel it carried would become a serious biohazard if it crashed in a populated area and would be a public relations disaster in any case. In contrast, highly enriched uranium is not highly radioactive before a reactor is first operated and so would be far less of a biohazard or public relations issue. Second, there would be some difficult technical problems to solve with plutonium. Plutonium has an exceptionally low thermal conductivity among metals, e.g., about 0.0674 W/cm-Kelvin in its alpha solid phase. Plutonium also melts at 640C so it would be a **liquid** metallic fuel at the much higher temperatures needed for this design. Operating with liquid fuel would confer benefits. For instance the usual materials issues concerning burn-up dependent fuel damage would completely disappear since it is impossible to damage a liquid. Xenon is continuously released from the liquid instead of being trapped in a solid fuel matrix, so reactivity transients are reduced. Fuel meltdown is no longer a disaster, strongly stable fuel temperature coefficients result, and refueling concepts are greatly simplified. However, an enormous amount of engineering development would be needed.

Although there is not much experience base in using liquid metallic plutonium fuel, there is some. The Los Alamos Molten Plutonium Reactor Experiment (LAMPRE) used 25 kg of liquid Pu^{239} confined in small tantalum capsules immersed in flowing sodium coolant [Kiehn 1957], [Swickard 1959], [Hanks and Taub 1960], [Glasstone 1960-3], [Harper and Garde 1981]. This experimental reactor operated up to its design thermal power level of one megawatt during the years from 1959 through 1963. LAMPRE's technical reports are available on-line from the Los Alamos library. However in my opinion, to obtain higher heat flux using molten plutonium would likely require developing pumped circulation of the fuel in order to overcome its low thermal conductivity. It would certainly be necessary to evaluate the effects of high temperature on the compatibility of confining materials (e.g., tantalum) with liquid plutonium and its fission products. In summary, to do an adequate job designing with molten plutonium would require far more resources than can be summoned at this time.

Given that the fuel will not be plutonium, U^{235} mixed with the least amount feasible of U^{238} is the next best choice. Pure enriched uranium metal would ideally provide the highest possible atom density of this fissile nuclide, thus minimizing the size of a critical configuration. Unfortunately, at temperatures 668C and 776C uranium's solid crystal structure changes from alpha to beta to gamma phases. In addition, pure uranium metal melts at 1130C Each of these phase changes represents a dramatic change in the fuel's size and shape which must be avoided in a reactor.

For the EBR-I reactor, 10% molybdenum alloyed with metallic uranium stabilized the uranium's gamma phase down to room temperature, so that changes between solid phases could be avoided [Glasstone 1967]. Melting issues were avoided by not operating at temperatures approaching the uranium alloy's melting point temperature, which was

elevated above pure uranium's melting point temperature due to the molybdenum component. However, EBR-I operation revealed that the solid metallic fuel became physically disrupted by fission products to the point of unusability in only a short period of operation. The highest burn up obtained from any of the EBR-I fuel was only 5.5 MWd/kgU and the average obtained before the fuel became unusable was only 3 MWd/kg U [Grossbeck 2005]. These low burn-up levels were disappointingly small fractions of the approximately 1000 MWd/kg U fissile energy content that would theoretically be available if the uranium fuel were highly enriched in the U-235 isotope.

Subsequently, most uranium-bearing fuel used in power rectors has been in the form of the ceramic material, uranium oxide (UO_2). This material has a lower thermal conductivity (0.036 W/cm-Kelvin) than metallic uranium, but a higher operating temperature is possible since its melting point temperature is 2800C. Most importantly, UO_2 fuel is routinely usable through fission burn up exposures of 60 MWd/kg U without any serious materials problems [Grossbeck 2005].

Uranium nitride (UN) fuel was chosen for use in the SP100 space nuclear power design project and was also adopted for the Project Prometheus reactor design project to power an ion rocket system. UN fuel can also be operated at high temperatures since the high temperature to be positively avoided with it is its decomposition temperature, also 2800C. Use of UN fuel helps with heat transfer since its thermal conductivity is about 0.25 W/cm-Kelvin. It is of interest to the breeder reactor community because of its good thermal and neutronic properties. Because of its small experience base as fission fuel its burn up capability is not reliably known, but published data indicate UN likely equals or even exceeds the burn up capability of UO_2 fuel. However, the most important reason UN is chosen here for the present nuclear engine design is that its uranium has a higher atom density than uranium has in UO_2 so a smaller critical configuration can be engineered using UN. Uranium nitride has a *uranium* content of 13.5 g/cm^3 whereas the *uranium* content of uranium dioxide is only 8.8 g/cm^3. Thus, the chosen fuel for the present nuclear engine design is UN using highly enriched uranium.

Literature searches for technical information on uranium nitride fuel found the information to be relatively sparse compared to more conventional fuel, e.g., UO_2. However, papers found on-line from non-US sources did provide a substantial amount of physical properties information for UN fuel, e.g., [Vatulin et al 2003].

4.9 Candidate Moderator and Shielding Materials

Appendix G contains a survey of candidate moderator and shielding materials along with apparently useful descriptive information for each. It was prepared by consulting various reference publications. The question this list addresses is which materials have engineers found in the past to be useful for projects with similarities to this nuclear engine design, and what properties must be considered in their application. The purpose of compiling this list was to avoid wasting time investigating the use of nonoptimal materials.

4.9.1 Guidance From Publications About Radiation Shielding/Neutron Moderation

According to Reactor Shielding [Schaeffer 1973, 450-2]

> "Weight rather than cost is generally the principal design criteria for mobile reactor systems. The emphasis, therefore, shifts to the improved shielding provided by the so-called exotic materials used in layered combination.
>
> The approach in designing minimum weight shields for reactor systems usually is to select the most efficient of the neutron shielding materials compatible with the design and to combine this with the most efficient of the gamma ray shielding materials rather than to select a single bulk attenuator medium. These two component materials are usually layered in some manner that yields minimum weight by providing the proper balance between primary and secondary radiations and geometric effects on material volume.
>
> The efficiency of a neutron attenuator can be correlated to its hydrogen density, and the efficiency of a gamma ray attenuator can be correlated to the total density of the material. Thus minimum-weight shield component materials are such combinations as lead plus polyethylene, tungsten (wolfram) alloys plus lithium hydride, or depleted uranium plus titanium hydride."

4.9.2 Gamma Ray Shielding

Figure 4-6 shows the narrow beam gamma attenuation per unit distance for selected materials, with its vertical axis the tenth layer thickness and its horizontal axis the gamma ray photon energy. Among these materials, gamma attenuation is greatest in uranium, then tungsten, then lead, then iron. If we divide these data curves by density to obtain a measure of gamma ray attenuation per unit line integral of shielding **mass**, the sequence changes. Uranium is still best on a mass basis but is closely followed by lead. Tungsten is not as effective as lead on the mass basis. Iron's attenuation on the mass basis is close to tungsten's, but they cross at different energies.

4.9.3 Neutron Shielding

Neutron shielding is complicated since it involves both slowing and absorbing the neutrons. For the relatively high energy neutrons of the fission spectrum, absorption cross sections are negligible and scattering cross sections are small. For such fast neutrons, inelastic scattering is frequently more significant than elastic scattering. As neutron energy is reduced elastic scattering becomes stronger and absorption cross sections become significant. For thermal spectrum neutrons, absorption cross sections can be quite large. Thus, conventional neutron shielding for fission spectrum neutrons first augments inelastic scattering if possible in order to reduce neutron energies to the range where elastic scattering becomes effective, then it provides elastic scattering, then finally absorbs the resulting slow neutrons.

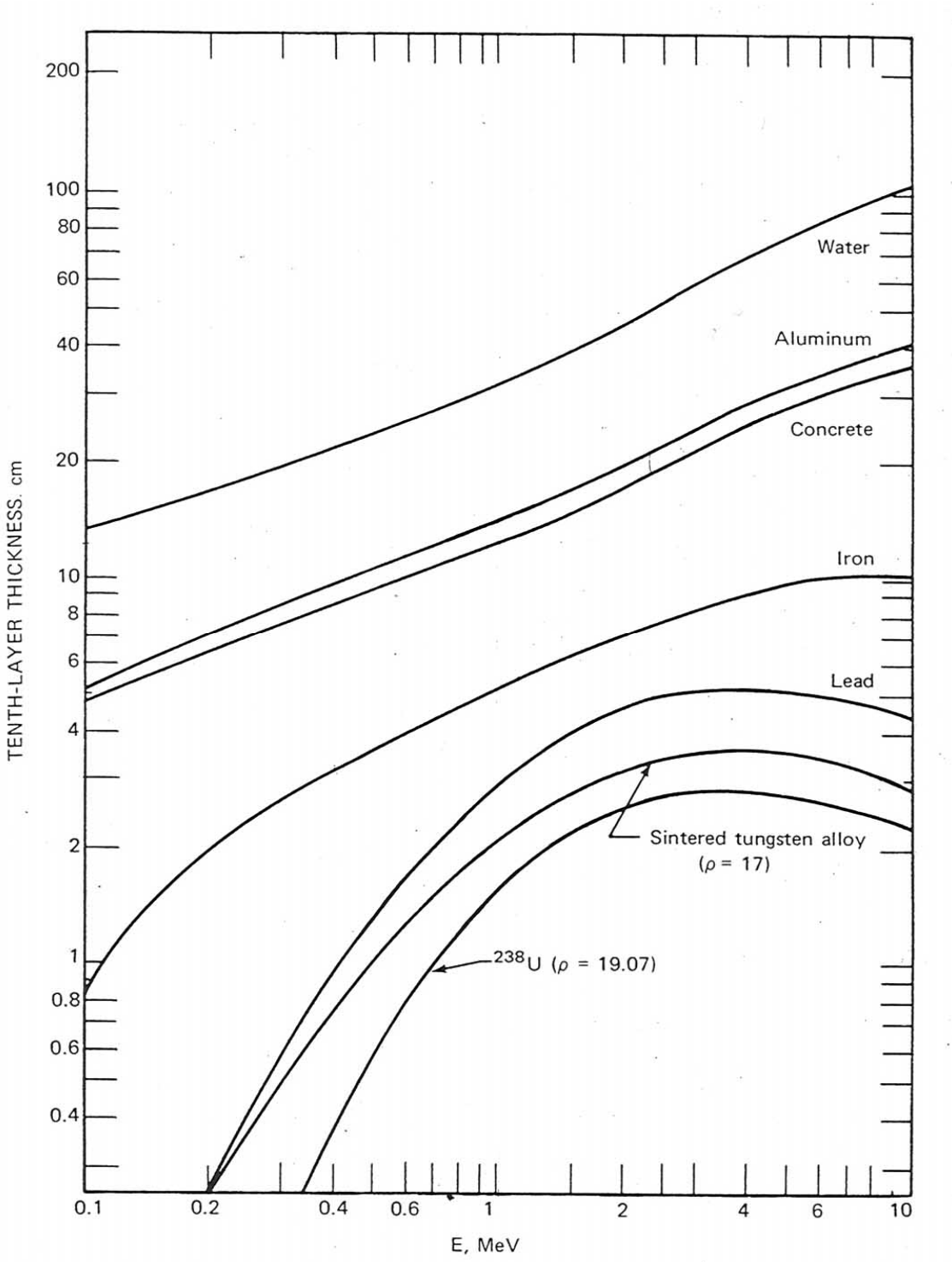


Figure 4-6: Gamma Ray Attenuation with Distance (From Reactor Shielding)

Figure 4-7 shows the neutron shielding effectiveness of several popular materials. A simplified but very approximate model is provided by the "removal" cross-section concept. If a beam of fission spectrum fast neutrons passing through a candidate scattering material is followed by sufficient hydrogenous material, i.e. 50 cm if the hydrogenous material is water, an effective removal cross section can be calculated by comparing exiting neutron beams with and without the candidate material present. The resulting experimental removal cross sections have validity for cases where their *macroscopic* cross section times their thickness does not exceed about 5 [Shultis and Faw 1996, 279-82]. The Table 4-3 microscopic removal cross sections were extracted from the same reference, then macroscopic values added using standard densities of solids.

Macroscopic removal per unit density provides a comparison of effectiveness for parts of the shield where the neutrons are not spatially diverging very much. For a small reactor surrounded by a spherical shield, that would approximate the situation in the outer parts of the shield. Ranking the listed materials by their macroscopic removal per unit *density*, the highest is lithium followed by beryllium, then boron, then carbon, then aluminum, then iron, then copper and nickel, then zirconium, etc.

The macroscopic removal cross section provides a comparison of effectiveness for the innermost parts of a shield where spatial divergence is strong. Ranking the listed materials, tungsten is highest by a clear margin followed by a near-tie between uranium, iron, copper, and nickel. After a gap these are followed by boron and beryllium, then after another gap by lead, then bismuth, then aluminum, etc., with lithium taking last place.

Figure 4-8 shows the effect of an inner removal shell of tungsten on the attenuation provided by a following shell of hydrogenous materials, in this case LiH.

Neutron absorption with most materials is effected by an (n,γ) reaction, so there is some gamma photon production within most neutron shielding materials. This is illustrated in Figure 4-9. Capture gamma photons in natural LiH, which are not plotted here, are far less than these curves because neutron absorption in Li^6 competes with radiative capture in the hydrogen.

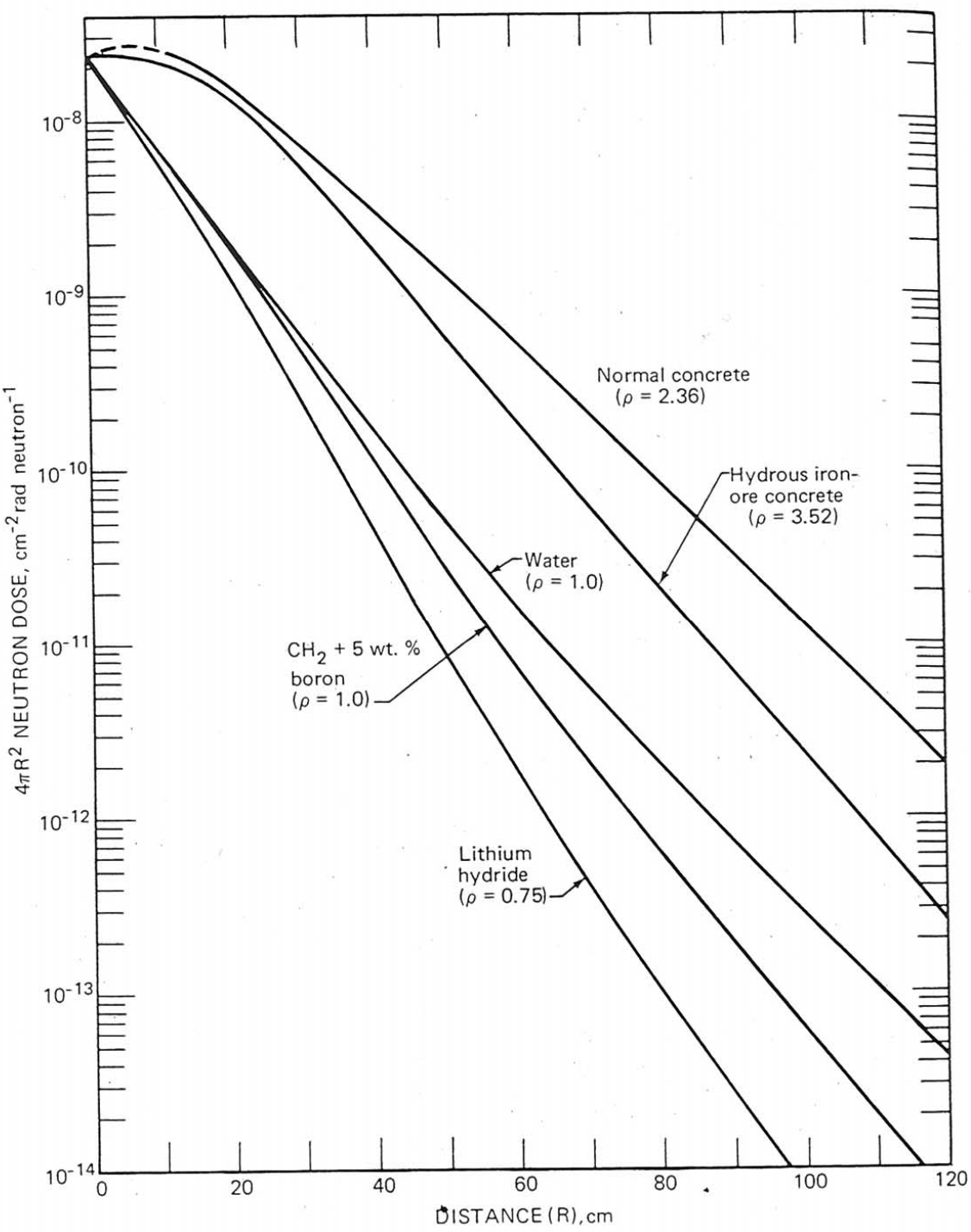


Figure 4-7: Fission NeutronAttenuation Of Selected Materials (from Reactor Shielding)

Table 4-3 : Measured Neutron Removal Cross Sections Of Selected Materials

Material	Microscopic Removal Cross-section σ_r (b/atom)	Atomic Weight A (g/mole)	Pure Solid Material Density ρ (g/cm^3)	Macroscopic Removal Per Unit Density $=0.6022\sigma_r/A$ ($g*cm^2$)	Macroscopic Removal Cross Section $=0.6022\rho\sigma_r/A$ (cm^{-1})
Aluminum (Al)	**1.31**	**26.98**	**2.70**	**0.0292**	**0.07895**
Beryllium (Be)	**1.09**	**9.01**	**1.85**	**0.0715**	**0.1323**
Bismuth (Bi)	**3.49**	**208.98**	**9.79**	**0.0100**	**0.0985**
Boron (B)	**0.97**	**10**	**2.37**	**0.0584**	**0.1384**
Carbon (C)	**0.81**	**12**	**2.15**	**0.0406**	**0.0874**
Chlorine (Cl)	**1.20**				
Copper (Cu)	**2.04**	**63.55**	**8.96**	**0.0193**	**0.1732**
Fluorine (F)	**1.29**				
Iron (Fe)	**1.98**	**53.85**	**7.87**	**0.0221**	**0.1743**
Lead (Pb)	**3.53**	**207.20**	**11.3**	**0.0103**	**0.1159**
Lithium (Li)	**1.01**	**6.94**	**0.534**	**0.0876**	**0.0468**
Nickel (Ni)	**1.86**	**58.69**	**8.90**	**0.0191**	**0.1699**
Tungsten (W)	**3.36**	**183.84**	**19.3**	**0.0110**	**0.2124**
Uranium (U)	**3.60**	**238**	**19.1**	**0.0091**	**0.1740**
Zirconium (Zr)	**2.36**	**91.22**	**6.52**	**0.0156**	**0.1015**

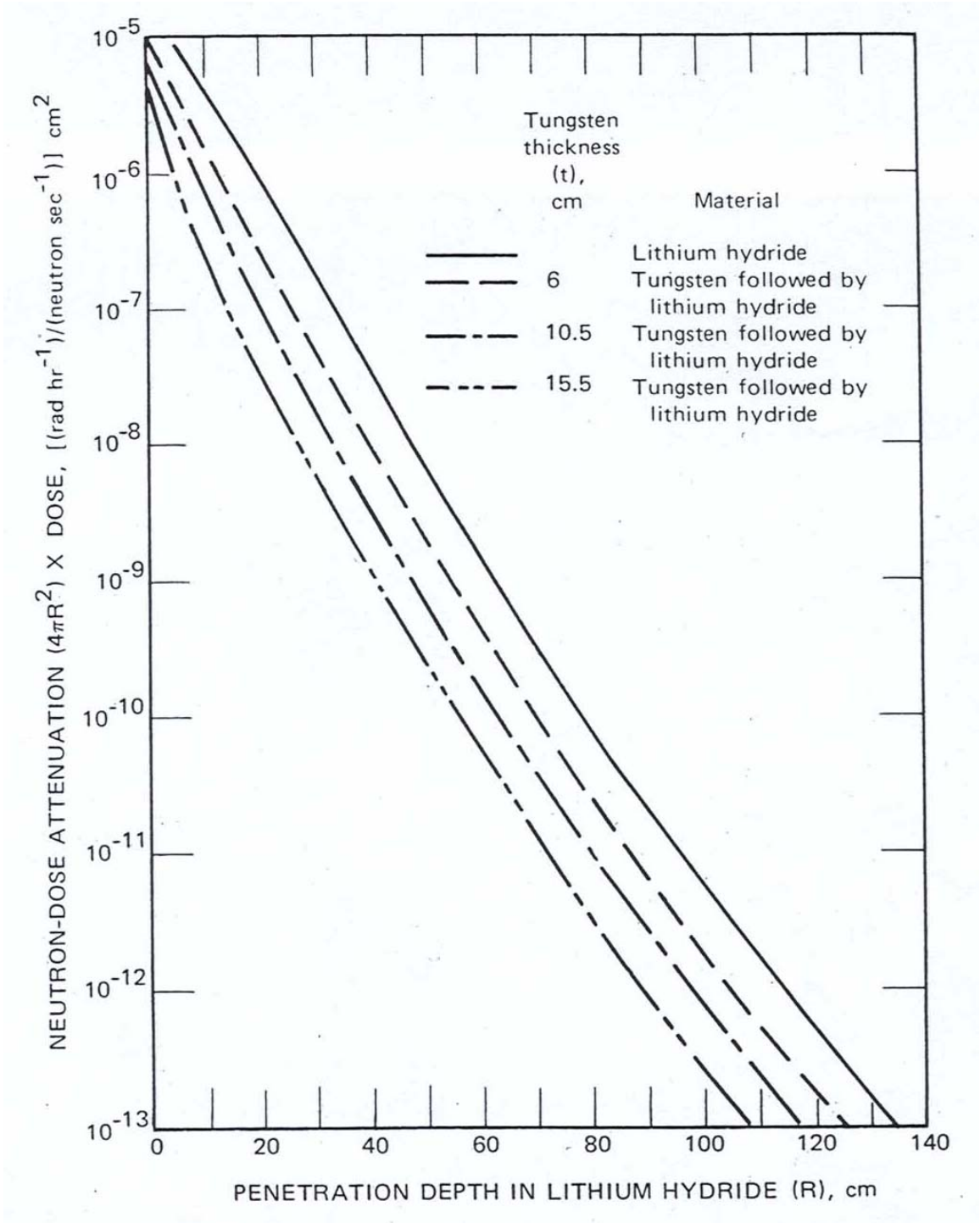


Figure 4-8: Tungsten & LiH Neutron Removal (from Reactor Shielding)

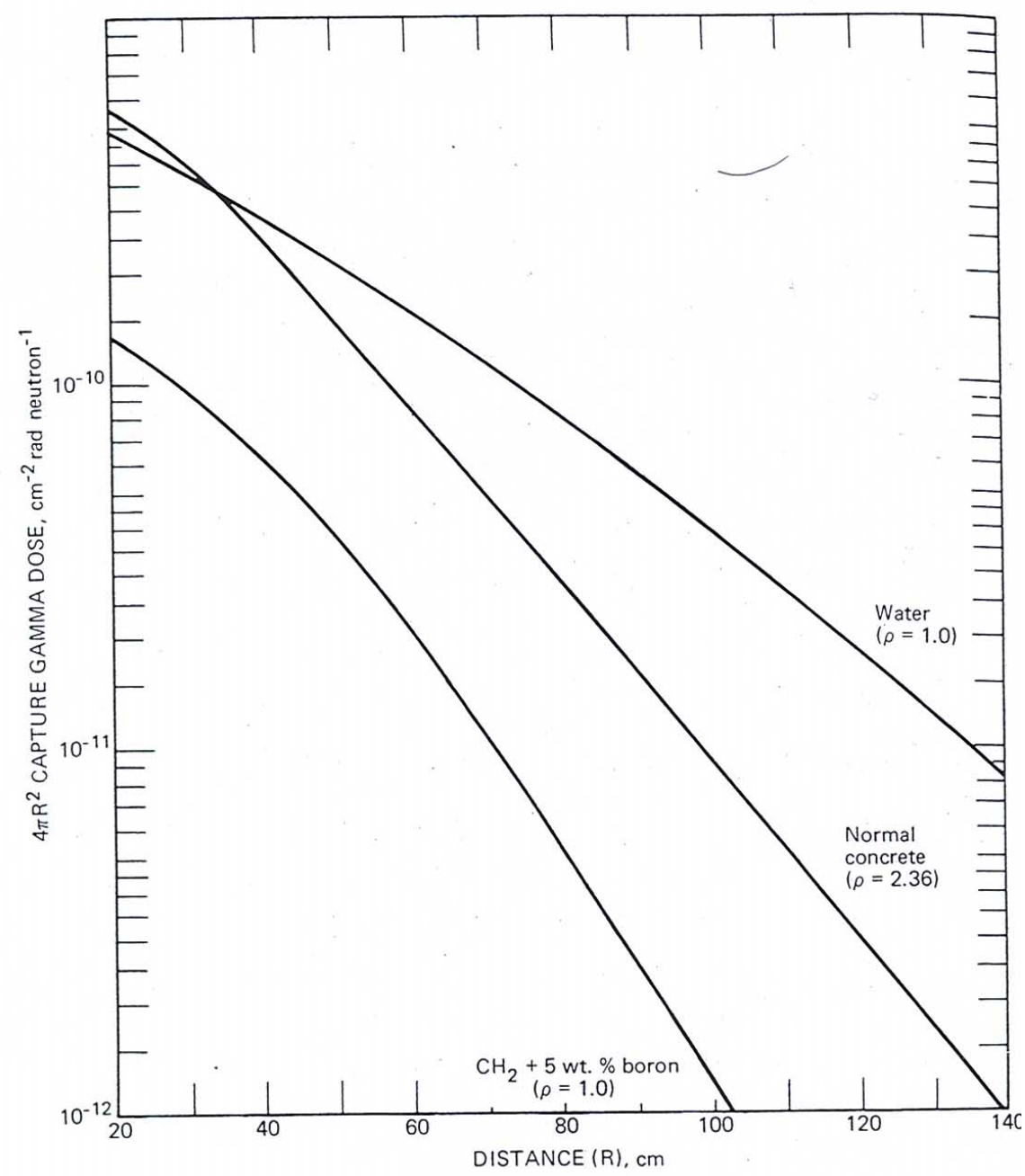


Figure 4-9: Neutron Capture Gamma Ray Production In Shielding Materials (from Reactor Shielding)

CHAPTER 5
5 INITIAL DISCRETE ORDINATES CRITICALITY STUDIES

5.1 Summary

How small can a power reactor be if it is otherwise suitable for a nuclear engine on Mars? More precisely, how small can the volume of its fissioning zone be, consistent with requirements for heat removal? Does the smallest reactor have a thermal neutron spectrum, a fast neutron spectrum, or is it intermediate between slow and fast? These questions motivated the initial criticality studies of the present chapter with the goal of developing an understanding of how geometry and materials placement affect criticality.

The discrete ordinates computer program used for these criticality studies was XSDRNPM, which is the RSICC's present-day successor to the legacy 1-D discrete ordinates code, ANISN. As stated in the XSDRNPM portion of the SCALE5 manual,

> "Development of the XSDRN program started in the mid-1960s. The goal was to develop a program that would combine features from the GAM-II, ANISN, and THERMOS programs in a more unified and general way than would be possible if one simply decided to use these codes individually .... The XSDRN program that embodied these features was released in 1969 ... The XSDRNPM module differs from XSDRN in several respects" including "It will perform coupled neutron-gamma calculations" and "It will perform an adjoint calculation, whereas this option was never provided in XSDRN" and also several other improvements.

However, XSDRNPM was not used directly. Instead, SCALE's control module for enhanced Criticality Safety Analysis Sequences, CSAS, invoked XSDRNPM after first invoking both the BONAMI program to perform self-shielding calculations for nuclides having Bondarenko data associated with their cross sections, and the NITAWL-III program to perform Nordheim Integral Treatment self-shielding corrections for nuclides having resonance parameters, as per the SCALE5 manual's CSAS section [SCALE 2005].

All computer runs of the present chapter investigated spherical configurations with an external vacuum boundary. Many studied varying the amount or placement of neutron moderator materials. Because of the importance for radiation shielding of using a thick tungsten layer close to a fission neutron source (as mentioned in the previous chapter), most of the criticality configurations investigated included a tungsten shell.

Throughout these investigations it was assumed that the uranium in the uranium nitride fuel used is "Weapons Grade" Highly Enriched Uranium (HEU), which is defined as having the isotopic composition shown in Table 5-1 [Stacey 2001, 232].

Table 5-1: Uranium Isotopic Composition of UN Fuel

Isotope	Percent
U-234	0.12
U-235	94.00
U-238	5.88

5.2 Criticality In The Limit Of A Solid Fuel Sphere

CSAS runs were first made to assess criticality of a sphere of pure Uranium Nitride fuel, both for a bare sphere and for a sphere embedded in a thick tungsten shell. Although these are not realistic power reactor configurations, they establish the limiting critical sizes when no moderator is present. A sample "input deck" for one of the cases follows.

```
=CSAS1X         parm=size=4000000
test
44GROUPNDF5                 multiregion
UN      1  DEN=14.6  1.0000  1400  92235 94.00  92238 5.88  92234 0.12 END
W       2  DEN=19.4  1.0000  1273 END
END COMP
spherical vacuum reflected 0 end
1 8.0
2 100.0 end zone
END
```

Results from these CSAS runs are summarized in Table 5-2 and plotted in Figure 5-1. Interpolation of these data leads to estimates of the k-effective=1 critical fuel sphere radii as being 11.08 cm for the bare case and 7.64 cm for the thick tungsten shell case. Apparently the tungsten shell provides radial "reflector savings" of 11.08-7.64=3.44 cm.

5.3 Criticality Consistent With A First-Cut Thermal Hydraulic Model

A high-power reactor cannot use a sphere of solid fuel since in that configuration the thermal power generated could not be steadily removed. There must be provisions for circulating reactor coolant through spaces provided within the fuel assembly. These spaces depend on the power level and thermal-hydraulics, but they directly affect criticality, so they are important here.

In this section they are first estimated, then sets of CSAS computer runs with a spherical configuration were performed to investigate how criticality depends on reactor size and on the volumetric fractions assigned alternatively to the fuel system vs. the moderator. The run sets thus investigated different neutron spectra ranging from thermal to fast. In all cases examined the following were assumed:

(1) spherical symmetry with two or three layers,
(2) external vacuum boundary,
(3) thick outermost shell is pure tungsten,
(4) inner sphere is a homogeneous mixture of UN fuel, W-Re clad, Li-7 coolant, and a moderator, solid density Be, or liquid density Li-7H.
(5) Each scan set varies "x", the inner sphere's volumetric moderator fraction.

Table 5-2: UN Sphere Criticality .

Radius of UN Fuel Sphere Using HEU (cm)	K-effective If Fuel Sphere Is Bare	K-effective w/ Thick Tungsten Shell (1 meter)
5.0	0.47146	0.68255
6.0	0.56672	0.81422
7.0	0.65968	0.93268
8.0	0.74947	1.03822
9.0	0.83543	1.13185
10.0	0.91712	1.21484
11.0	0.99426	1.28848
12.0	1.06673	1.35386

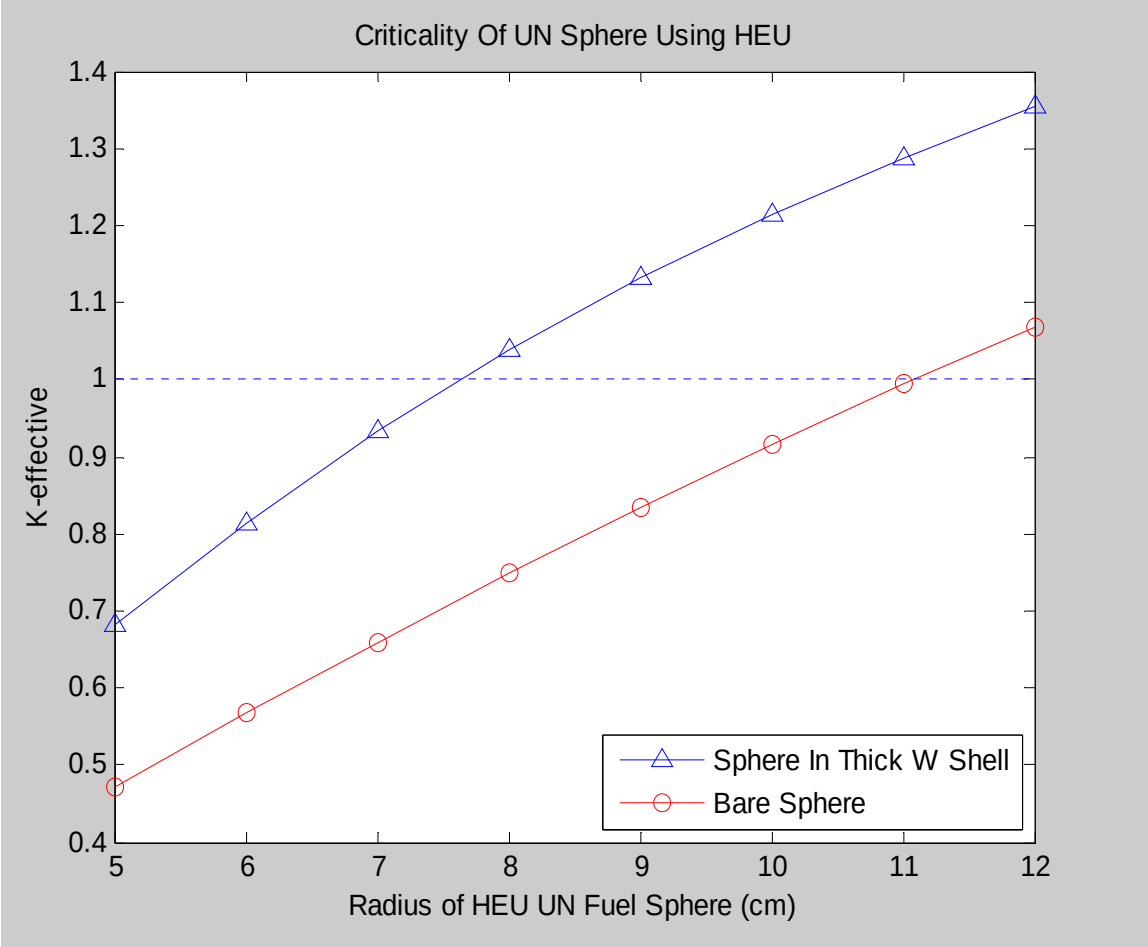


Figure 5-1: Criticality of HEU UN Fuel Sphere

5.3.1 A First-Cut Model of Thermal Hydraulics And Of Other Considerations

The SP100 project was pursued from 1978 to 1993 to design a reactor for space developing 100 kw electric output power continuously for a 7 year operating lifetime. Since it was to operate at high temperature with UN fuel and lithium coolant, similar to the present nuclear engine design, information about its design parameters was consulted. A significant source of SP100 technical information was found on-line at the internet site of the University of Wisconsin's Fusion Technology Institute, http://fti.neep.wisc.edu/fti , which has made the SP100 design a central feature of the on-line course notes for one of its courses.

The designed SP100 reactor's 2.4 MW thermal power was transferred to its lithium coolant at 1375 Kelvins reactor exit temperature. Of that thermal power, 2.3 MW would be rejected through a true radiator at 790 Kelvins as waste heat, while 0.1 MW would be converted to electric power. Thus, the thermal efficiency which the SP100 design team settled on after fifteen years of design activity significantly funded by the US government was about **4%**. This is far less than the 25% typical of vehicle engines on Earth or the 33%-40% range of efficiencies typical of stationary terrestrial power plants. However, low efficiency thermoelectric energy conversion was chosen because it was a reliable and proven technology.

Based on the SP-100 information, the global density of UN uranium is about 13.5 g/cubic cm, and the maximum possible uranium fuel mass could not have exceeded 102.2 kg. Now, the thermal power claimed for SP100 was 2.4 MWt for 7 years, which works out to be 6136 MW-days. Therefore, the SP100 design must have been counting on a burn up level from its UN fuel of at least

(6136 MW-days/0.1022MT)=60,040 MWd/MTU =60.04 kwd/g U

As a somewhat arbitrary first-cut design parameter, I required a reactor core of the present nuclear engine design to operate continuously at 1 MW thermal power for the 18 month typical duration of a manned expedition's surface stay on Mars, without needing refueling due to fuel material damage. The integrated thermal power would be

(1 MW)(30 d/month)(18 month/surface stay)=540 MWd/surface stay.

To get 540 MWd from UN fuel with the same level of burn up that apparently was assumed by the SP100 project, we would need a total uranium mass of

(540 MWd)/(60,040 MWd/MTU)= 8.994 kg.

To package 8.994 kg of uranium as UN of uranium density 13.5 g/cm^3 requires 8994/13.5=666 cm^3 of UN. This volume could be obtained by 19 plates, each 10 cm-square, separated by 1 mm gaps, with each plate containing a 3.52 mm thickness of fuel meat and each having cladding and liner thicknesses of 0.25 mm and 0.13 mm respectively to match the cladding and liner thicknesses of the SP100's fuel design. A stack of the 19 plates separated by 1 mm spacing gaps would be 9.93 cm thick, so the stack would approximate a 10 cm cube.

I assumed these fuel plates were arranged in the staggered configuration of Figure 4-5 so that they can provide a high ΔT low flow rate thermal hydraulic design. Including lithium duct end regions beyond the 19 stacked plates, that provides 20 lithium heating regions each 10 cm in length, for a total of 2 meters heated duct length. Lithium duct cross-section dimensions would be 1 mm X 10 cm, giving a total cross-sectional lithium flow area of 1 square centimeter (1 cm^2).

It would be preferable from the standpoint of maximizing thermal conversion efficiency to increase the reactor core exit temperature of the lithium coolant above the SP100 project's 1375 K. The published reference discussed earlier [Zinkle and Ghoniem 2000] indicates that this temperature could be increased to 1643 K without adverse materials interactions between the lithium and suitable confining materials. However, some caution is warranted since such a temperature increase would imply increasing the operating temperature of the UN fuel. It is unknown to the author what the side effects might be of operating the fuel at its corresponding increased temperature.

Assuming we want the lithium to heat compressed martian air exiting the Open Brayton Cycle's turbocompresser at 580 Kelvins and raise its temperature to 1270 Kelvins, the most we would be willing for lithium to be simultaneously cooled in a low flow-rate thermal hydraulic design would be from its 1375 Kelvins reactor exit temperature to about 685 Kelvins, which would maintain a constant temperature drop across an ideal counter-flow liquid-to-gas heat exchanger of

1375-1270=685-580=105 Kelvins.

Therefore for this first-cut calculation the temperature rise of the lithium coolant is set at

ΔT =1375-685=690 Kelvins.

To thermally transfer 1 MW with 690 Kelvins ΔT requires a lithium volumetric flow rate of 0.694 liters/second. The speed would then be (0.694 liters/second)/(1 cm^2)=6.94 m/sec. Duderstadt and Hamilton 1976 lists a typical speed of LMFBR liquid sodium coolant as 4 m/sec, so 6.94 m/sec in lithium appears plausible.

With a 1 mm thick slot 10 cm wide, the duct's hydraulic diameter is (4)(.01)(.001)/(.02+.002)=1.82 mm. The Reynolds number of the duct flow would then be (6.94m/s)(0.00182m)/(kinematic viscosity) where the kinematic viscosity varies from 7.444E-7 m^2/s @690K to 4.514E-7m^2/s @1370K. Picking 6E-7 as a representative kinematic viscosity value we get a Reynolds number of 21,051, which is well into the turbulent regime. Moody's diagram then gives the friction factor as 0.0255, so the total pressure drop of coolant flowing through the reactor is approximately

ΔP=0.5(0.0255)(2m/0.00182m)(500kg/m^3)*(6.94m/s)^2 =

=337409.8 Pa

=48.9 psi.

This pressure difference would be divided across the plates. In some locations the pressure difference across a plate could be as much as 5 psi. Some mechanical design attention might need to be given to ensuring plates have enough internal stiffness to

withstand such pressure drops. Perhaps some internal stiffening ribs may be appropriate, or perhaps plates could be corrugated.

Mechanical pumping power consumed in the reactor core would be small compared to the 1 MW of thermal power transferred, i.e.,
PumpingPower=(337409.8 Pa)(0.694E-3m^3/s)=234.16 watts.

The Prandtl number of a fluid is defined [Gebhart 1971. 5], as

$$\mathrm{Pr} = \frac{c_p \mu}{k}$$

where c_p is the specific heat at constant pressure, μ is the viscosity, and k is the thermal conductivity. For liquid lithium these have mid-temperature-range values of

$$c_p = (0.995 cal/g - K)(4.1868 J/cal)(1000 g/1 kg) = 4165.8 \text{ J/kg} - \text{K}$$

$$\mu = 28.75E - 5 \text{ Pa} - \text{s}$$

$$k = 40.3 \text{ W/m} - \text{K}$$

Combining this gives the mid-temperature-range Prandtl number of liquid lithium as being

$$\mathrm{Pr} = \frac{c_p \mu}{k} = \left(\frac{(4165.8)(2.875E - 4)}{40.3} \right) = 0.0297$$

The Lyon-Martinelli correlation characterizing heat transfer involving flowing liquid metals subject to a constant wall heat flux is given by [Duderstadt and Hamilton 1976, 486 .]

$$Nu \approx 6.3 + 0.030 Pe^{0.8}$$

$$Pe \equiv \mathrm{Re}\,\mathrm{Pr}$$

Plugging in the pertinent values here gives

$$Nu \approx 6.3 + 0.030((21051)(0.0297))^{0.8} = 11.475$$

Then the heat transfer coefficient is given by

$$h_s = \frac{k}{D_h} Nu = \frac{40.3 \text{ W/m - K}}{0.00182 \text{ m}} (11.475) = 254089 \text{ W/m}^2\text{K}$$

$$\approx 25.4 \text{ W/cm}^2\text{K}$$

With 19 plates sharing 1 MWt, each is producing 52.6 kW. Since each plate has 200 cm^2 of external cooled surface, the average heat flow is 52.6kW/200=263W/cm^2. Then, the average film temperature drop from the clad to the nearby bulk fluid temperature is

$$\Delta T_{film} = \frac{263}{25.4} = 10.4 \text{ K}$$

With the fuel thickness being 3.52 mm, the average power density in the fuel is 263W/cm^2/0.352cm=747 W/cm^3. The thermal conductivity of the UN fuel is 0.25 W/cm-K. Then

$$0.25\frac{d^2T}{dx^2}+747=0$$

where x is the distance from the fuel plate central plane in cm. Thus

$$\frac{d^2T}{dx^2}=-2988$$

A symmetric solution is

$$T=T_{center}-(46.278)\left(\frac{x}{0.176}\right)^2$$

which shows that the maximum temperature rise within the fuel is only 46.278 C.
In summary, this flat plate fuel scheme appears fine for a first simple look at criticality consistent with thermal hydraulics. Of the 10 cm cube of fuel plates, cladding, and lithium forming this core, the first-cut volume fractions of Table 5-3 would prevail.

The radius of a sphere with the same volume as this 10-cm cube is R=6.231 cm.

5.4 Criticality Using Beryllium As A Moderator

The material volume fractions of Table 5-3 were adopted for these runs, with the change being that tungsten was adopted as cladding. For Be moderator cases, volumetric fractions within the core were as per the following formulae for different values of x.

Be VF=x
UN VF=0.660(1-x)
Li-7 VF=0.197(1-x)
W VF=0.094(1-x)
Re VF=0.049(1-x)

Actual volume fraction cases run are listed in Table 5-4.

For different size fuel assemblies this implies the U235 content of Table 5-5.

CSAS runs were made using these volume fractions for the inner sphere representing a reactor core and using pure tungsten (W) for the outer spherical shell. The outer radius of the tungsten spherical shell was in all cases set to 50 cm. The inner sphere's radius is tabulated below, along with the resulting calculated k-effective values. Note that VF values of zero caused CSAS to generate erroneous results so minimum VF values were set to 0.0001 instead of zero.

Table 5-3: First-Cut Reactor Core Volume Fractions

Material	Volume (cm^3)	Volume Fraction
UN	668.8	0.660
Rhenium Liner	49.4	0.049
Cladding	95.0	0.094
Lithium Coolant	200.0	0.197
Total	1013.2	1.000

Table 5-4: Volume Fractions In 1st-Cut CSAS Runs Investigating Be As Moderator

Case	Material Volume Fractions				
	Be	UN	Li-7	W	Re
x=0	0.0000	0.6600	0.1970	0.0940	0.0490
x=0.1	0.1000	0.5940	0.1773	0.0846	0.0441
x=0.2	0.2000	0.5280	0.1576	0.0752	0.0392
x=0.3	0.3000	0.4620	0.1379	0.0658	0.0343
x=0.4	0.4000	0.3960	0.1182	0.0564	0.0294
x=0.5	0.5000	0.3300	0.0985	0.0470	0.0245
x=0.6	0.6000	0.2640	0.0788	0.0376	0.0196
x=0.7	0.7000	0.1980	0.0591	0.0282	0.0147
x=0.8	0.8000	0.1320	0.0394	0.0188	0.0098
x=0.9	0.9000	0.0660	0.0197	0.0094	0.0049
x=0.95	0.9500	0.0330	0.0099	0.0047	0.0025
x=0.98	0.9800	0.0132	0.0039	0.0019	0.0010
x=0.99	0.9900	0.0066	0.0020	0.0009	0.0005
x=0.995	0.9950	0.0033	0.0010	0.0005	0.0002
x=0.999	0.9990	0.0007	0.0002	0.0001	0.0000

Table 5-5: Kilograms Of Uranium-235 Content In Cases Run

Moderator Vol. Fraction	Radius of Spherical Reactor Fuel Assembly (in Centimeters)				
	R=6.2	R=10.0	R=15.0	R=20.0	R=25.0
0.000	8.534	35.807	120.849	286.457	559.486
0.100	7.680	32.226	108.764	257.811	503.538
0.200	6.827	28.646	96.679	229.166	447.589
0.300	5.974	25.065	84.594	200.520	391.640
0.400	5.120	21.484	72.509	171.874	335.692
0.500	4.267	17.904	60.425	143.228	279.743
0.600	3.414	14.323	48.340	114.583	223.794
0.700	2.560	10.742	36.255	85.937	167.846
0.800	1.707	7.161	24.170	57.291	111.897
0.900	0.853	3.581	12.085	28.646	55.949
0.950	0.427	1.790	6.042	14.323	27.974
0.980	0.171	0.716	2.417	5.729	11.190
0.990	0.085	0.358	1.208	2.865	5.595
0.995	0.043	0.179	0.604	1.432	2.797
0.999	0.009	0.036	0.121	0.286	0.559

A sample input deck follows. Table 5-6 lists the calculated k-effective results.

```
=CSAS1X         parm=size=4000000
test
44GROUPNDF5                 multiregion
BE      1           0.0001  1400 END
UN      1  DEN=14.6  0.6600  1400  92235 94.00  92238 5.88  92234 0.12 END
LI-7    1  DEN=0.45  0.1970  1350 END
W       1  DEN=19.4  0.0940  1400 END
RE      1  DEN=19.4  0.0490  1400 END
W       2  DEN=19.4  1.0000   973 END
END COMP
spherical vacuum reflected 0 end
1 6.2
2 50.0 end zone
END
```

These tabulated results are plotted as Figure 5-2. Inspection of this plot reveals that for a homogeneous spherical reactor assembly with a fixed size, the highest criticality value of k-effective occurs when all the reactor core volume is devoted to the mixture of fuel/clad/coolant and none of the volume is devoted to beryllium moderator. Thus, this plot fails to show any criticality advantage of using beryllium as a moderator, in terms of reducing the reactor core size. The plot also shows that for any particular volume percent of beryllium moderator, k-effective increases monotonically with the size of the sphere.

The k-effective data can also be plotted against fissile fuel mass as shown in Figure 5-3. Here, the incrementally beneficial effect of the Be moderator is visible in the R=20 cm and R=25 cm cases as a local increase in the k-effective curves for low fissile mass data points. This partially demonstrates the point that **use of a moderator in the proper configuration can reduce the critical fissile mass**. However, as shown this enhancement is not sufficient to allow k-effective=1 criticality even for the R=25 cm case using beryllium as the moderator. A sphere larger than R=25 cm would be needed to include sufficient Be moderator and fuel in the assumed configuration to achieve criticality with a soft neutron energy spectrum.

Next some 3-layer configurations employing pure beryllium in one layer were examined. Initially, beryllium shells were substituted for the *differences* between core radii listed in Tables 5-5 and 5-6. Figure 5-4 compares these three-layer cases with the two-layer cases already examined in which no beryllium was used. Since the cases already examined included R=6.2 cm and R=10.0 cm cases, the first 3-layer case examined substituted pure Be for their overlap region. Putting in a 10-6.2=3.8 cm thick spherical shell of Be around a 6.2 cm radius reactor sphere, and keeping in place the thick external tungsten layer out to R=50 cm yielded a k-effective value of 0.68266. This was only a slight increase from the 0.63090 value for the 6.2 cm radius sphere and nowhere near the 0.97377 value which resulted from using a 3.8 cm thick spherical shell of additional fuel/clad/coolant instead of Be. Thus in this case substituting Be for the fuel/clad/coolant mixture *hurts* criticality

Table 5-6: K-effective Values From CSAS Runs for 2-layer Homogeneous Spherical Reactors with Be Moderator in 1m Diameter Tungsten Sphere

Beryllium	K-effective Values Vs Radius of Spherical Reactor Fuel Assembly				
Vol. Fraction	R=6.2 cm	R=10.0 cm	R=15.0 cm	R=20.0 cm	R=25.0 cm
0.000	0.63090	0.97377	1.27369	1.46388	1.59033
0.100	0.59691	0.93711	1.24033	1.43478	1.56484
0.200	0.56029	0.89656	1.20232	1.40073	1.53422
0.300	0.52061	0.85140	1.15880	1.36077	1.49752
0.400	0.47732	0.80067	1.10856	1.31365	1.45345
0.500	0.42964	0.74302	1.04992	1.25753	1.40012
0.600	0.37642	0.67635	0.98023	1.18956	1.33460
0.700	0.31582	0.59712	0.89485	1.10473	1.25174
0.800	0.24444	0.49828	0.78430	0.99256	1.14067
0.900	0.15439	0.36133	0.62219	0.82342	0.97043
0.950	0.09489	0.25834	0.49274	0.68640	0.83262
0.980	0.04765	0.16442	0.37735	0.57946	0.74295
0.990	0.02721	0.11546	0.31891	0.54104	0.73398
0.995	0.01497	0.07759	0.26096	0.49443	0.71510
0.999	0.00350	0.02370	0.11037	0.26290	0.44600

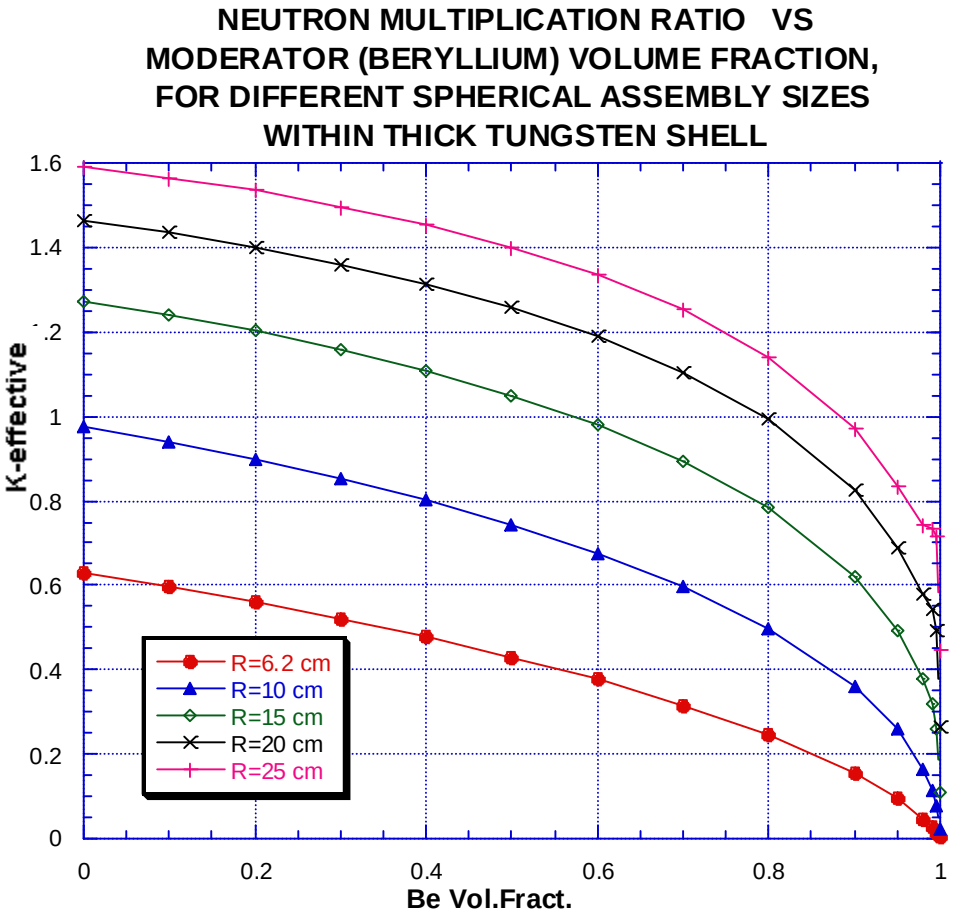


Figure 5-2: K-effective Vs. Beryllium Moderator Fraction For Different Core Sizes

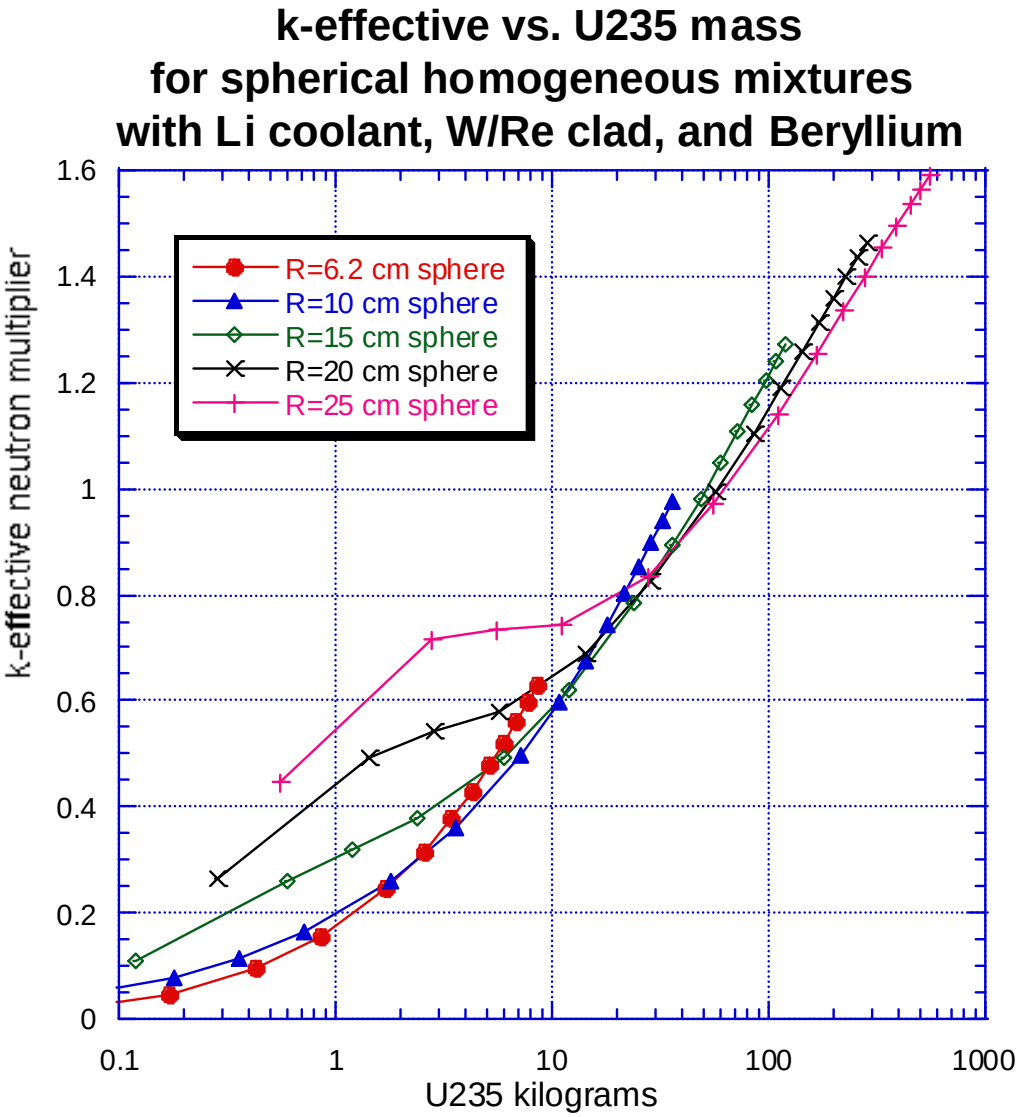


Figure 5-3: K-effective Vs U-235 Content for Different Be Fractions And Core Sizes

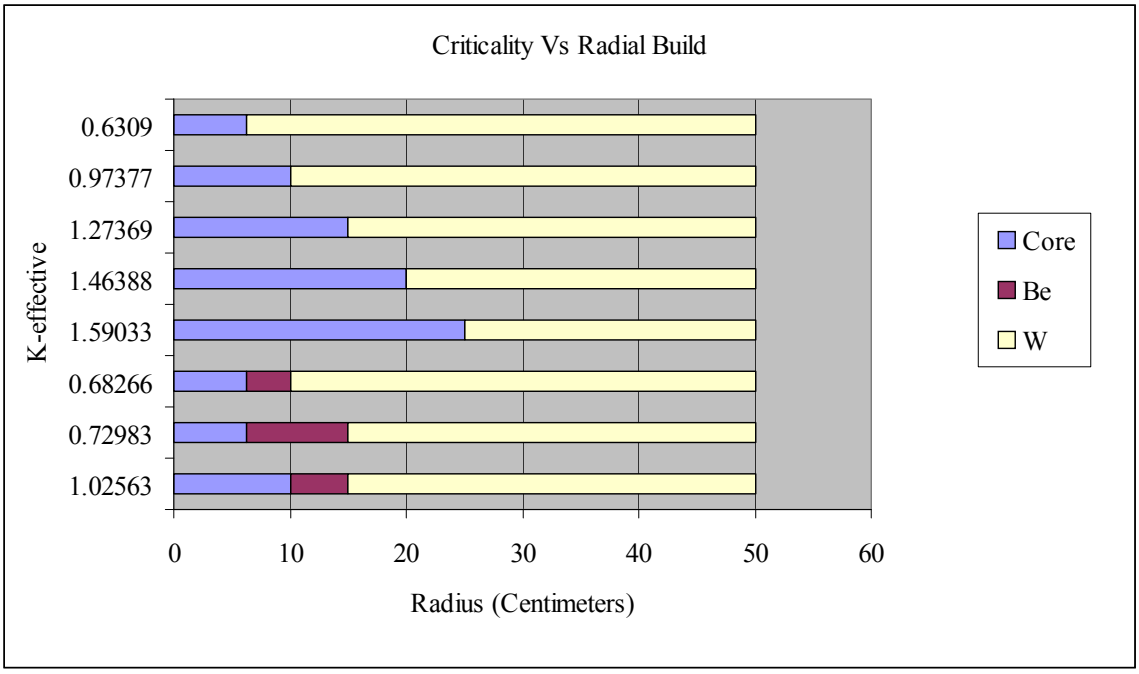


Figure 5-4: Comparisons of 2 and 3 Layer CSAS Runs Using Beryllium

For the next 3-layer run the 6.2 cm fuel/clad/coolant radius is kept and the thickness of the Be shell is increased to 8.8 cm so that the pure Be shell would extend all the way to a radius of 15 cm, thus matching the 15 cm fuel/clad/coolant radius of a previous 2-layer run. The resulting k-effective value became 0.72983, far less than 1.27369 of that previous 2-layer run in which the core extended out to R=15 cm. Thus, substituting a shell of 8.8 cm thick Be for an equal volume of fuel/clad/coolant again *hurts* criticality.

Similarly, if the fuel/clad/coolant radius is 10 cm and a 5 cm thick Be shell surrounds it, the k-effective value is 1.02563. This is an increase over the 0.97377 value obtained without any Be, but is much less than the 1.27369 value obtained by using fuel/coolant/clad mixture to R=15 cm instead of beryllium. Thus, in all cases, sacrificing an outer shell portion of the reactor core in exchange for pure beryllium *hurts* criticality.

Next one more variation not shown in Figure 5-4 was investigated, trading off fuel/clad/coolant vs. Be at the spherical reactor's center. Converting the innermost 1 cm radius from fuel/clad/coolant mixture to pure beryllium slightly *decreased* k-effective from 0.97377 to 0.97305. Further increasing the radius of the central Be sphere to 2 cm further reduced k-effective to 0.96851. Thus, all results were consistent. *Any* trading of fuel/clad/coolant reactor core material for beryllium resulted in reducing k-effective.

Summarizing these results, every trade of Be for core material resulted in an adverse effect on criticality and thus on reactor size. This was true whether the Be was spread evenly throughout the core volume, concentrated as an outer shell, or concentrated in the center. Although using Be may reduce the mass of fissile material needed, the total volume of that fissile material plus the beryllium is increased whenever Be is substituted. This suggests that **the optimum amount of Be to use as a moderator may be zero**.

On the other hand, a more favorable interpretation is possible. In each 3-layer case where pure Be was added it can alternatively be viewed as substituting for tungsten. In each such case, k-effective *increased* slightly over the k-effective value for a core of the same size surrounded by tungsten alone. Thus, surrounding a core by a beryllium shell within a tungsten shell may help, permitting a smaller core than if only tungsten were used.

5.5 Criticality Using Lithium-7 Hydride As Moderator

For cases where the moderator is lithium-7 hydride, the density of lithium hydride assumed was 0.55 g/cm^3, its **liquid** state density at 700 Ca discussed in Appendix G. The densities of the two components within the Li7H are then calculated as follows:

Li (0.55)(7/8)=0.48125=0.45*1.06944
H (0.55)(1/8)=0.06875

Then the composition of the fuel/clad/coolant is given by the following recipe:

H DEN=0.06875 VF=x
UN VF=0.660(1-x)
Li-7 DEN=0.45 VF=0.197(1-x)+1.06944x
or VF=0.197+0.87244x
W VF=0.094(1-x)
Re VF=0.049(1-x)

The material volume fractions used in these first-cut runs investigating use of 7-lithium hydride as a moderator are as summarized in Table 5-7. The U235 content of Table 5-5 applies also to these cases. Table 5-8 lists the computed k-effective values from these CSAS runs and Figure 5-4 plots those results versus the Li^7H moderator's core volume fraction.

Apparently Li^7H is a much better moderator than Be, even in its low density liquid form. This is evidenced by the upward bumps at high Li^7H volume fractions visible in the Figure 5-5 curves plotted for the larger size reactor cores but which are not visible in the corresponding Figure 5-2 curves for the Be moderator. This larger moderator effect with Li^7H than with Be is also visible in the Figure 5-6 plot of k-effective vs. fissile material mass, in a way not very much in evidence in Figure 5-3.

The Figure 5-6 plot clearly shows that for the larger spheres the use of liquid LiH as moderator can achieve criticality with only a small mass of fissile material in the assumed mix proportions of uranium nitride, W/Re clad, and Li coolant. Indeed, only 5 kg of uranium with R=21 cm would suffice.

However, it is also clear that even for LiH, **the smallest size sphere to contain fuel plus moderator results from using no moderator at all**. This situation would not hold if there existed a sufficiently better moderator. Indeed, computer runs in which the LiH density is erroneously increased by a factor of ten predict the opposite result. However, no material with such high moderating power exists. Thus, the optimality of a fast-neutron-spectrum reactor in achieving small size within a gamma shield appears to hold true in practice.

On the other hand, it is also necessary to provide shielding for the neutrons leaking from the fuel assembly which in turn requires that they must first be slowed by moderator material external to the fuel assembly. That situation was not modeled in the above CSAS runs. Some of the slowed neutrons might leak back into the fuel and enhance the fission rate. Thus, it is still possible that the fastest neutron spectrum may not be optimal.

Three-layer CSAS runs in the same geometric arrangements previously run for beryllium were also examined. The results are shown in Figure 5-7.

Figure 5-7 shows the situation with an outer shell of Li^7H is different from the corresponding situation with beryllium. In each of the three cases examined, replacing tungsten by Li^7H caused k-effective to decrease whereas it increased when the replacing material was beryllium as in Figure 5-3.

Table 5-7: Material Volume Fractions Used In First-Cut CSAS Runs Investigating Li^7H As Moderator

Case	Volume Fractions				
	H (DEN= 0.06875)	UN	Li-7 (DEN= 0.45)	W	Re
x=0	0.0000	0.6600	0.1970	0.0940	0.0490
x=0.1	0.1000	0.5940	0.2842	0.0846	0.0441
x=0.2	0.2000	0.5280	0.3715	0.0752	0.0392
x=0.3	0.3000	0.4620	0.4587	0.0658	0.0343
x=0.4	0.4000	0.3960	0.5460	0.0564	0.0294
x=0.5	0.5000	0.3300	0.6332	0.0470	0.0245
x=0.6	0.6000	0.2640	0.7205	0.0376	0.0196
x=0.7	0.7000	0.1980	0.8077	0.0282	0.0147
x=0.8	0.8000	0.1320	0.8950	0.0188	0.0098
x=0.9	0.9000	0.0660	0.9822	0.0094	0.0049
x=0.95	0.9500	0.0330	1.0258	0.0047	0.0025
x=0.98	0.9800	0.0132	1.0520	0.0019	0.0010
x=0.99	0.9900	0.0066	1.0607	0.0009	0.0005
x=0.995	0.9950	0.0033	1.0651	0.0005	0.0002
x=0.999	0.9990	0.0007	1.0686	0.0001	0.0000

Table 5-8: K-effective Values From CSAS Runs for 2-layer Homogeneous Spherical Reactors with Li^7H Moderator in 1m Diameter Tungsten Sphere

Li^7H	Radius of Spherical Reactor Fuel Assembly (in Centimeters)				
Vol. Fraction	R=6.2	R=10.0	R=15.0	R=20.0	R=25.0
0.000	0.63107	0.97395	1.27379	1.46386	1.59022
0.100	0.60662	0.94210	1.23190	1.41387	1.53422
0.200	0.58128	0.90821	1.18729	1.36113	1.47566
0.300	0.55412	0.87148	1.13967	1.30570	1.41482
0.400	0.52451	0.83148	1.08886	1.24750	1.35164
0.500	0.49180	0.79777	1.03465	1.18644	1.28606
0.600	0.45521	0.73996	0.97711	1.12282	1.21855
0.700	0.41360	0.68777	0.91707	1.05829	1.15129
0.800	0.36501	0.63172	0.85839	0.99910	1.09223
0.900	0.30303	0.57344	0.81484	0.96826	1.07108
0.950	0.25400	0.53579	0.80530	0.98199	1.10221
0.980	0.18923	0.47070	0.77192	0.98049	1.12599
0.990	0.13694	0.39026	0.69391	0.91676	1.07650
0.995	0.08903	0.28992	0.56621	0.78609	0.95025
0.999	0.02452	0.09760	0.23001	0.36029	0.47107

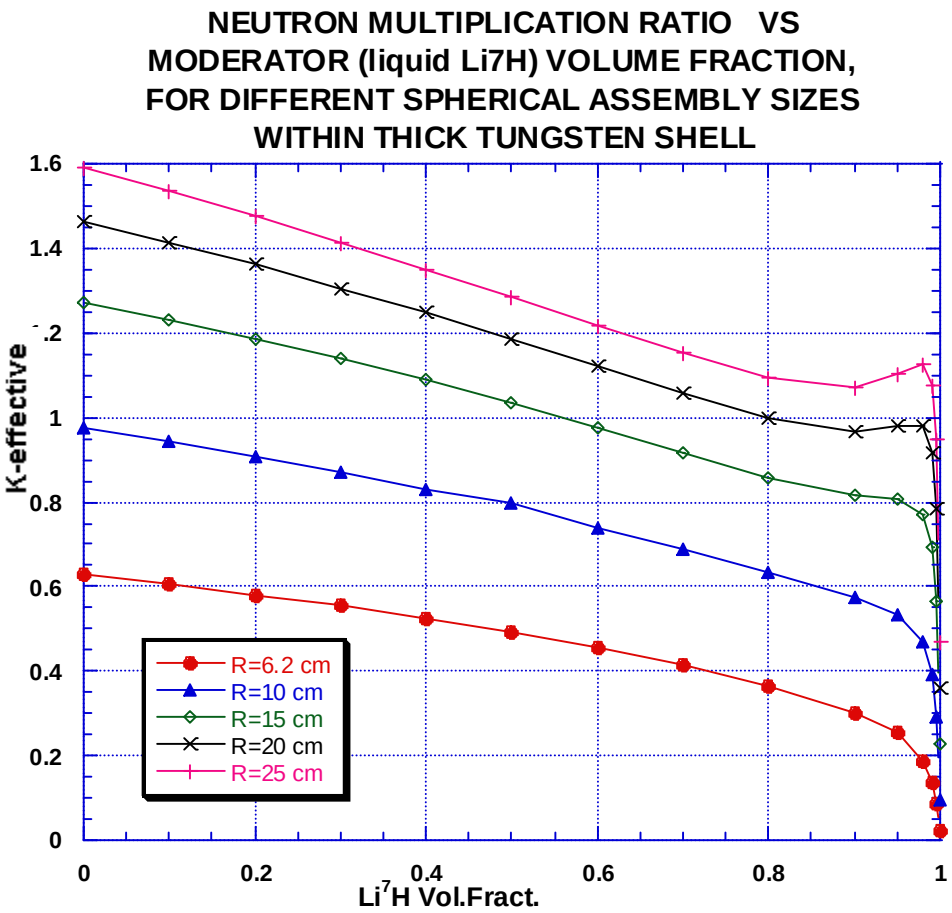


Figure 5-5: K-effective Vs. Li^7H Moderator Fraction For Different Core Sizes

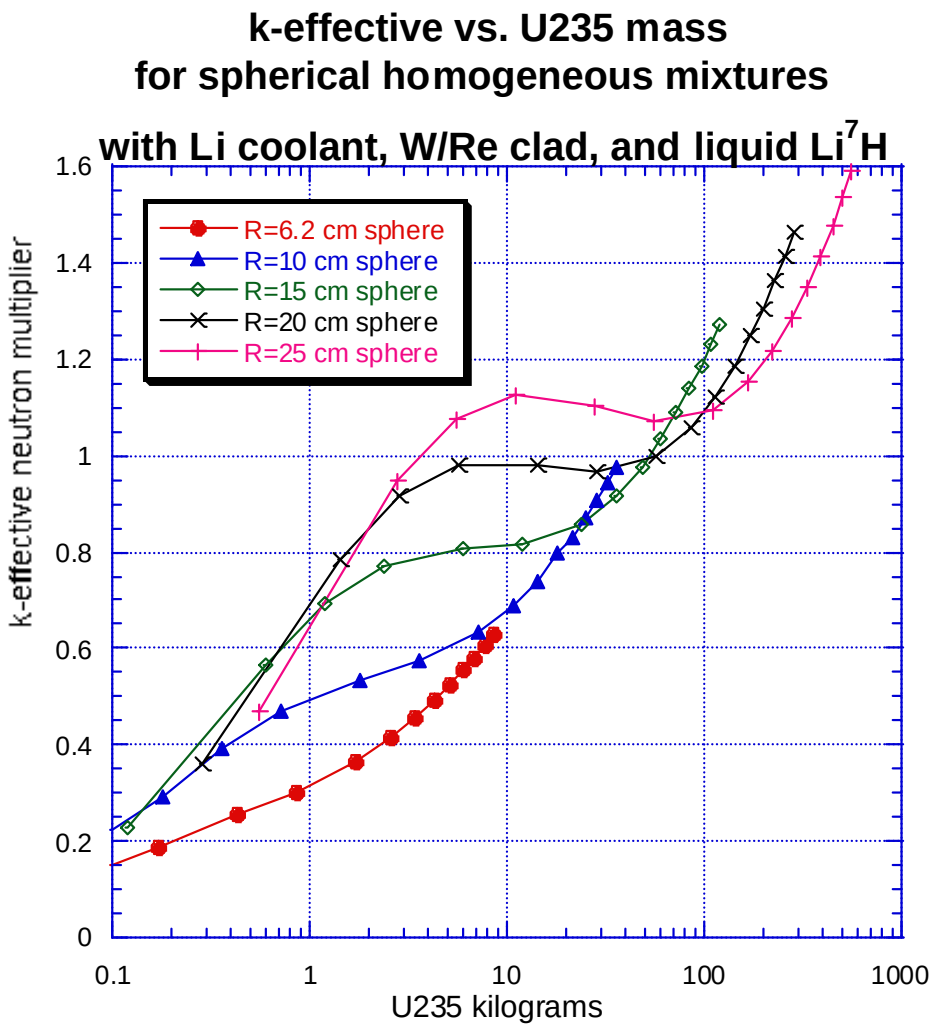


Figure 5-6: K-effective Vs U-235 Content for Different Li^7H Fractions And Core Sizes

5.6 Making Sense Of 3-Layer Run Results

The 3-layer criticality results may initially seem paradoxical. On the one hand, Li^7H is known to be a better neutron moderator than Be, both from consulting references and from comparing Figures 5-1 5-2, 5-4, and 5-5 On the other hand, adding a Be shell helps criticality while adding a Li^7H shell hurts criticality.

However, neutron moderation is not the only phenomenon involved here. Material shells also act as **neutron reflectors**. Some materials, e.g., beryllium, are better neutron reflectors than tungsten, while other materials are not as effective as tungsten in reflecting neutrons. To investigate this further, additional 3-layer runs were made using alternatively vacuum shells and shells of 1100 Kelvins density 10.05 g/cm^3 molten lead.

Results are plotted in Figure 5-8. For most of the cases presented, beryllium is the best shell material to increase criticality, followed by tungsten. A vacuum gap shell hurts criticality more than any material shown here. Lithium hydride and lead are intermediate between tungsten and a vacuum, gap. For the R=6.2 cm core, Li^7H helped criticality more than Pb, while for the R=10 cm case Pb helped criticality more than Li^7H.

Some 4-layer cases were also run to examine the possibility of combining Be with Li^7H. Here, the same core mixture was modeled to 6.2 cm, then Be to 10 cm, then Li^7H to 15 cm followed by W to 50 cm. The resulting k-effective value was 0.67137, which is slightly less than the 0.68277 results with Li^7H replaced by W as shown at the bottom of Figure 5-7. Expanding the Li^7H shell thickness so that it extends from 10 to 20 cm radius increases k-effective to 0.68073. Reversing the sequence so that Li^7H extends to 10 cm while Be extends from 10 to 20 cm also results in k-effective being 0.68073. With Li^7H extending from 6.2 to 10 cm and Be from 10 to 15 cm, the rest being W, the resulting k-effective is 0.69894. None of these runs showed any benefit from Li^7H.

5.7 Criticality With A Second-Cut Thermal-Hydraulic Model

In the previous criticality investigations, it was assumed that the uranium nitride was limited to at most 0.66 of the volume of the reactor core. This was done to provide enough space for Li coolant to flow between 19 plates each of which was assumed in advance to be 10 cm by 10 cm, so that the total heat transfer surface would be 3800 square centimeters. This in turn was chosen to limit average surface heat flux to about 265 W/cm^2 in the nominally 1 MWt reactor case. However, the criticality result was not consistent with those assumptions. It showed criticality would *almost* be achieved for a 10 cm radius sphere having that assumed homogeneous composition. Now, a 10 cm

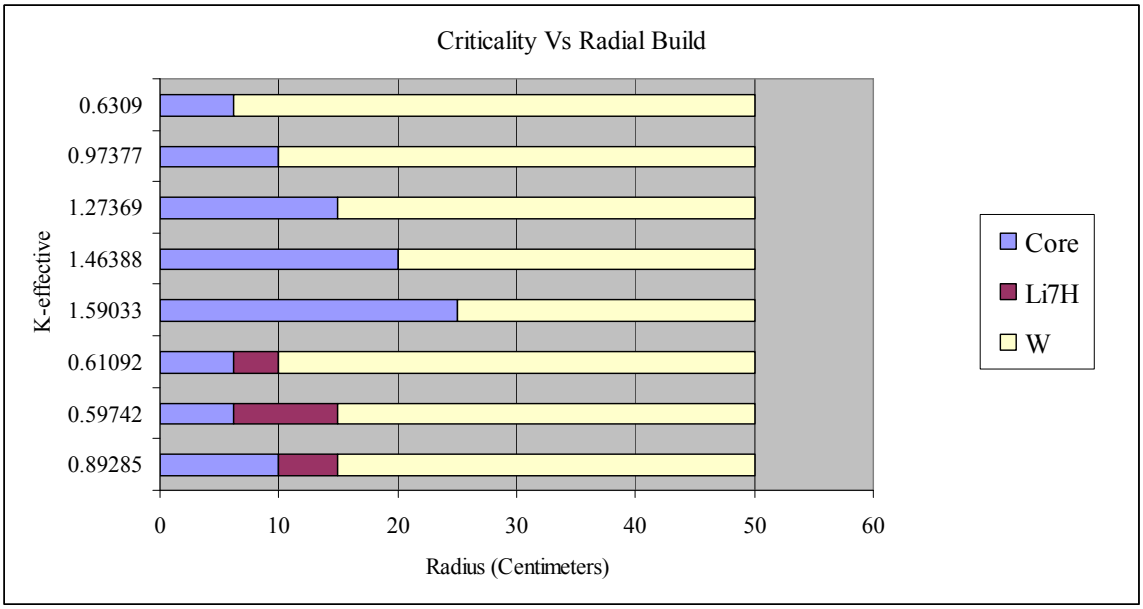


Figure 5-7: Comparisons of 2 and 3 Layer CSAS Runs Using Li^7H

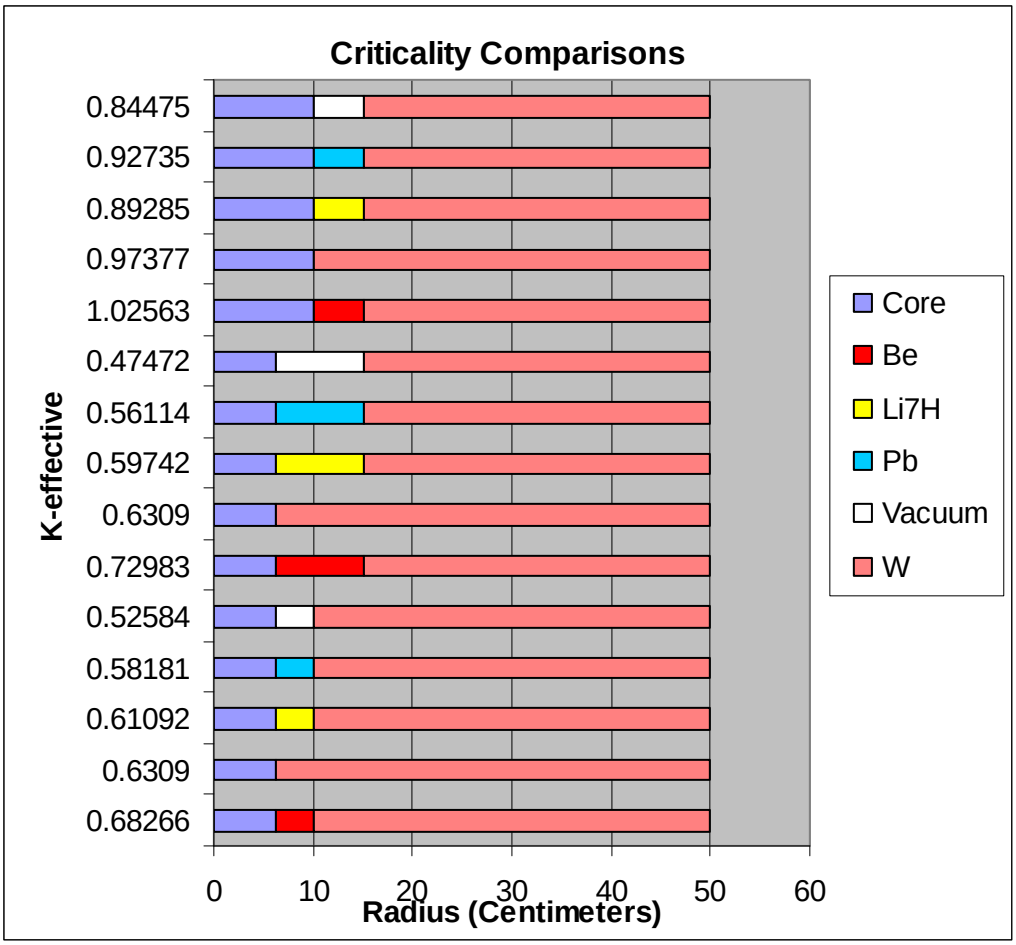


Figure 5-8: Comparison of 3-Layer Criticality Results Using Be,W,Pb,$Li^{7}H$,Vacuum

radius sphere has the same volume as a cube 16.12 cm on each side. Employing the same fuel plate thickness and spacing, that implies 31 plates each with 260 cm^2, giving a total surface area of 16,111 cm^2, i.e., over four times the surface area assumed in first cut runs.

It would be better to increase fuel plate thickness to increase the fuel volume fraction. To this end, the unchanging dimensional parameters were chosen as listed in Table 5-9. Keeping these parameters constant and requiring the entire assembly of fuel plates to be in the form of a cube, Table 5-10 resulted. With total reactor power also kept constant, thermal hydraulic parameters external to the fuel plate surfaces remain almost constant across these different cases. However, the temperature rises internal to the fuel are different for different cases.

Selected CSAS criticality results assuming a thick tungsten shell are shown in the right hand column of Table 5-11. Criticality k-effective=1 is exceeded first for the 14.142 cm cube, modeled for CSAS by R=8.7731 cm sphere of listed composition.

Considering the question of how the external neutron shield would impact these criticality estimates, it initially seemed plausible that a moderator, which is necessary in a neutron shield, might reflect some slowed neutrons back into the reactor core and that these could soften the neutron energy spectrum which would increase the average fission cross section. Anticipating that k-effective might increase as a result, the initial focus was on the 12-plate case instead of the 10-plate case. Using the 12-plate data, the volume fraction of the UN fuel was therefore set to 0.8287, the molten lithium coolant's VF to 0.1007, the tungsten's VF to 0.0465 and the Rhenium's to 0.0242. However, the resulting CSAS run outputs showed k-effective *decreasing* with the substitution of external moderator for tungsten. Results are shown in Table 5-12.

These results demonstrate that substitution of Li^7H moderator for tungsten in various shells surrounding a R=8.007 cm sphere of fuel/clad/coolant always reduces k-effective. This follows the pattern investigated in the previous section and thus should not be additionally surprising. However, the k-effective reduction is not great if there is a sufficiently thick intervening tungsten shell. The k-effective reduction is large for a 2 cm thick W shell, significant at 7 cm thick, small at 12 cm thick, and very small for a 17 cm thick W shell.

In the light of these results, the 10-plate case of Table 5-10 was accepted as appropriate for further study, assuming a reactor thermal power of 1 megawatt. With 10 plates sharing 1 MW, each dissipates 100 kW. Since each is 14.14 cm square its external surface area is 398.75 cm^2 which implies the average surface heat flux is 250.8 W/cm^2. With the fuel thickness being 1.2282 cm, the average power density in the fuel is $(250.8W/cm^{2)})/(1.2282\ cm)=204.2\ W/cm^3$. Since the thermal conductivity of the UN fuel

Table 5-9: Fixed Dimensional Parameters of Reactor Core

Plate arrangements for 4000 cm^2 total heat transfer area in cube, 1 mm flow channels, clad/liner thicknesses (mm)	
W clad thickness	0.25
Re liner thickness	0.13
Li flow channel width	1.00

Table 5-10: Parameters of Cubic Stacks of Fuel Plates With Constant Heat Transfer Area Constant Thickness of Coolant Duct, Clad and Liner

Number of plates	Fuel Plate Side (cm)	UN fuel thickness (cm)	UN fuel volume (cm^3)	UN fuel VF	Molten Lithium Coolant VF	Tungsten Clad (W) VF	Rhenium Liner (Re) VF	Equiv. Sphere Radius (cm)	Calc'd CSAS results k-eff
19	10.2598	0.3587	717.45	0.6643	0.1949	0.0926	0.0481	6.3647	
18	10.5409	0.4041	808.10	0.6900	0.1802	0.0854	0.0444	6.5391	
17	10.8465	0.4561	912.30	0.7149	0.1660	0.0784	0.0408	6.7286	
16	11.1803	0.5165	1033.04	0.7392	0.1521	0.0716	0.0372	6.9357	
15	11.5470	0.5871	1174.27	0.7627	0.1386	0.0650	0.0338	7.1632	
14	11.9523	0.6706	1341.18	0.7855	0.1255	0.0586	0.0305	7.4146	0.83202
13	12.4035	0.7704	1540.84	0.8075	0.1129	0.0524	0.0273	7.6945	
12	12.9099	0.8915	1782.99	0.8287	0.1007	0.0465	0.0242	8.0087	0.91948
11	13.4840	1.0407	2081.45	0.8490	0.0890	0.0408	0.0212	8.3648	0.96661
10	14.1421	1.2282	2456.43	0.8685	0.0778	0.0354	0.0184	8.7731	1.01685
9	14.9071	1.4692	2938.47	0.8870	0.0671	0.0302	0.0157	9.2476	
8	15.8114	1.7879	3575.85	0.9046	0.0569	0.0253	0.0132	9.8086	
7	16.9031	2.2244	4448.88	0.9212	0.0473	0.0207	0.0108	10.4858	

Table 5-11: CSAS Results: 12-Plates With External Moderator

Fuel assy Sphere radius (cm)	Tungsten Shell Outer Radius (cm)	Lithium-7 Hydride Shell outer Radius (cm)	Second Tungsten Shell Outer Radius(cm)	k-effective calculated by CSAS
8.0087	10	10.01	50	0.91903
8.0087	10	11	50	0.87768
8.0087	10	12	50	0.84900
8.0087	10	13	50	0.83040
8.0087	10	14	50	0.81816
8.0087	10	15	50	0.81005
8.0087	10	16	50	0.80467
8.0087	10	17	50	0.80112
8.0087	10	18	50	0.79882
8.0087	10	19	50	0.79736
8.0087	10	20	50	0.79646
8.0087	10	21	50	0.79593
8.0087	10	22	50	0.79564
8.0087	10	23	50	0.79551
8.0087	10	24	50	0.79546
8.0087	10	25	50	0.79547
8.0087	10	26	50	0.79551
8.0087	10	27	50	0.79556
8.0087	10	28	50	0.79561
8.0087	10	29	50	0.79566
8.0087	10	30	50	0.79570
8.0087	10	35	50	0.79578
8.0087	10	40	50	0.79568
8.0087	10	45	50	0.79516
8.0087	15	45	50	0.86917
8.0087	20	45	50	0.90398
8.0087	25	45	50	0.91514
8.0087	30	45	50	0.91829
8.0087	35	45	50	0.91912
8.0087	35	95	100	0.91912

is 0.25 W/cm-K, the 1D temperature distribution in the fuel is approximated by the following equation:

$$0.25\frac{d^2T}{dx^2} + 204.2 = 0$$

where x is the distance from the fuel plate central plane in cm. A symmetric solution is

$$T = T_{center} - (154)\left(\frac{x}{0.6141}\right)^2$$

which shows the temperature rise in the UN fuel based on averaged conditions is 154 C.

Additional CSAS runs were made for this case, with the volume-preserving spherical radius of the core 8.7731 cm and the volume fractions set to 0.8685 for the UN fuel, 0.0778 for the molten lithium-7, 0.0354 for the tungsten cladding, and 0.0183 for the rhenium liner. A satisfactory criticality value of k-effective=1.01645 was obtained for the following spherical dimensions (in cm):

Core, 0<R<8.77
W, 8.77<R<38.77
Li^6H, 38.77<R<48.77, at LiH solid density
B-10, 48.77<R<50
W, 50<R<55.

For this case CSAS lists the production/absorption ratio as 1.017E+00 matching k-effective to its indicated significant figure precision level and indicating almost complete absorption of the neutrons. Since at this point the criticality condition has been demonstrated for a configuration consistent with thermal-hydraulic cosiderations, these reactor core parameters were adopted to use as the starting point to investigate shielding configurations.

5.8 Schemes For Reactivity Control

Results of criticality investigations as discussed above lead to the adoption of a fast-spectrum reactor design since it apparently provides the physically smallest fissioning volume for thermal powers near 1 megawatt. However, without significant moderator action, such reactor designs must rely on other mechanisms to achieve stability in operating at a particular power level. For thermal reactors such as PWRs a stable moderator temperature coefficient is achieved by designing cores to be under-moderated and exploiting thermal expansion of the moderator material. For PWR's using low enrichment uranium, a doppler coefficient for neutron absorption in U238 provides another favorable mechanism promoting stability. Unfortunately, neither of these is an available control mechanism for a HEU-fueled fast spectrum reactor.

The suggestion from Nuclear Reactor Theory [Bell and Glasstone 1970, 453] is to provide *some* moderator action and to use U238's doppler coefficient for resonant neutron absorption to engineer a negative temperature coefficient, using the small fraction of neutrons having low enough energy to be affected. The reference states this approach would rely on the small fraction of neutrons in the energy band from 1 to 10 keV. This approach would require introducing U238 together with enough Be or Li^7H to provide sufficient neutron moderation that a negative temperature coefficient would result. Perhaps the U238, Be, and Li7H for this approach could be limited to a shell surrounding the reactor core so that the fissioning volume could remain small and thus comparatively easy to shield. However, this does not seem to be a promising approach for a fast reactor since only the tiny fraction of the neutrons that had slowed would be affected.

There are other ways to adjust reactivity. First, the text by Lamarsh and Baratta [Lamarsh and Baratta 2001], 359] points out that the cross section of the boron-10 neutron absorption reaction falls off more slowly with increasing neutron energy than most (n,γ) reactions and that it remains large enough at high energies to be used in the control rods of fast reactors. Second, schemes to change the physical size of the fuel assembly in order to change reactivity may be straightforward to implement given the rectangular shape of fuel plates (e.g., by sliding the left and right sides of Figure 4-5 away from or towards each other). Either of these approaches could be combined with temperature sensitive linkages to effect a negative temperature coefficient.

A different control approach is investigated in the present section. Results of CSAS runs discussed above show that criticality is strongly affected by the shell surrounding the reactor core which reflects neutrons back into the core. Review of the SP100 space nuclear power design reveals that its beryllium reflector was designed to permit radial sliding of portions of the reflector so that by moving the beryllium pieces radially outwards the k-effective value would be reduced. Such a control scheme could be adopted here. A beryllium inner shell layer followed by a gap could be included along with active feedback control laws linking measured temperature to radial movement of the solid pieces of Be comprising this shell.

However, an even more direct approach not relying on active feedback control would use as the neutron reflector a fixed volume shell of *liquid* material thermally coupled to the reactor core but mechanically connected to an external liquid expansion volume. Then if the temperature of that liquid shell were to increase along with the reactor's temperature, the expanding liquid would naturally flow out from the shell, reducing the shell's mass and its ability to reflect neutrons back into the core, thus reducing k-effective.

An initial issue to confront is the choice of which liquid shell material to use. The material that would theoretically provide the best thermal feedback coefficient based on its strong reflective performance is liquid beryllium. However, since beryllium's melting point temperature is 1560 Kelvins (i.e., 1277 C) which is within or near the range of operating temperatures envisioned for the nuclear engine reactor, this choice would

require dealing with the technical problems that would accompany frequent liquid/solid phase transitions and their volume changes. For instance, problems could arise due to partial freezing, blocking the pipes connecting the shell to its external expansion volume, then subsequent freezing of the liquid beryllium in its cavity might rupture its boundary.

The material best thermally coupled to the reactor is the reactor coolant itself, Li-7. A reactor design could easily route the coolant through a shell region where it would serve as a neutron reflector. Unfortunately, Li-7 may not be a very good neutron reflector material. However, it is investigated here. Another material to consider is molten lead with its liquid range from 600 to 2026 Kelvins which includes most conditions for the nuclear engine of the present design. Lead would also help provide radiation shielding.

In addition to engineering a stable temperature coefficient, there must also be a mechanism to adjust for reactor core reactivity changes. A fast reactor may not need to cope with strong reactivity changes from fission product transients since absorption cross sections are not large for high energy neutrons, but the core's reactivity will slowly diminish as a result of fuel burn-up and that effect must be addressed.

To roughly assess the effect of anticipated burn up on reactivity, an estimate was made of the remaining fuel enrichment after operating for the same 60 megawatt-days per kilogram that apparently was assumed by the SP100 design team for UN fuel. Assuming about 1 megawatt-day per gram fissioned and assuming that only the U235 component of HEU fissions, this burn up level would reduce the remaining U235 content from 94% to about 88% of the *original* mass of uranium. In terms of the 33.2 kg of uranium comprising the 10-plate case, this level of burn up implies cumulative production of 1,990 megawatt-days of heat and would be reached after 5.45 years of continuous operation at one megawatt. CSAS found that changing the U235 content from 94% to 88% while making no other changes (i.e., keeping the core radius 8.7731 cm and surrounding it with a tungsten shell out to R=50 cm) had the effect of reducing the calculated k-effective from 1.01664 to 0.98549, a total reactivity change of -0.03115.

A series of 3-layer CSAS criticality runs was made to determine the effects on criticality of inserting shells of different thicknesses and different materials. Each case examined modeled the 10-plate reactor core consistent with 1 MWt operation as the 1st layer, i.e., a 8.7731 cm radius sphere. Each case included as its 3rd layer a tungsten spherical shell extending out to R=50 cm. The 2nd layer was alternatively chosen as one of seven different materials including tungsten) and it extended from its inner radius at 8.7731 cm to alternatively different outer radii where also the 3rd layer began.. Results for the different layer #2 materials and for the different radii of the layer 2/layer 3 interface are summarized in Table 5-12 and Figure 5-8.

Figure 5-8 shows that the largest decrease in k-effective results from substituting B-10 for tungsten reflector material, but that substituting a vacuum yields almost as much decrease. These are followed in turn by Li-7, then by Li^7H, then by molten Pb. Substitution of Be for the tungsten actually ***increases*** k-effective.

Table 5-12: 3-Layer K-effectives Calculated By CSAS (Consistent With Second Cut 1 MWt Model)

Outer Radius Of Layer #2 (InnerShell) (cm)	Layer #2 (Inner Shell) Material						
	Be (1.8 g/cm^3)	W (same as 2-layer case)	Pb (10.05g/cm^3)	Li7H (0.48125 +0.06875 g/cm^3)	Li-7 (0.45 g/cm^3)	Vacuum Gap	B-10 (2.37 g/cm^3)
8.7731	1.01664	1.01664	1.01664	1.01664	1.01664	1.01664	1.01664
10	1.03301	1.01664	0.99637	0.99236	0.98100	0.96843	0.95675
11	1.04333	1.01664	0.98460	0.97078	0.95885	0.93873	0.92112
12	1.05192	1.01664	0.97541	0.95454	0.94058	0.91431	0.89288
13	1.05963	1.01664	0.96814	0.94284	0.92523	0.89387	0.87012
14	1.06714	1.01664	0.96237	0.93447	0.91217	0.87658	0.85149
15	1.07486	1.01664	0.95777	0.92855	0.90097	0.86178	0.83612
16	1.08293	1.01664	0.95410	0.92447	0.89126	0.84898	0.82327
17	1.09135	1.01664	0.95116	0.92181	0.88277	0.83790	0.81218
18	1.10003	1.01664	0.94884	0.92023	0.87532	0.82818	0.80302
19	1.10882	1.01664	0.94701	0.91947	0.86874	0.81961	0.79518
20	1.11761	1.01664	0.94558	0.91930	0.86293	0.81203	0.78846
30	1.19069	1.01664	0.94261	0.92476	0.82927	0.76712	0.75393
40	1.23146	1.01664	0.94464	0.92632	0.81183	0.74395	0.74171
49	1.25002	1.01664	0.93878	0.92626	0.79088	0.72122	0.73354
50	1.25116	1.01664	0.93748	0.92623	0.78812	0.72005	0.73280

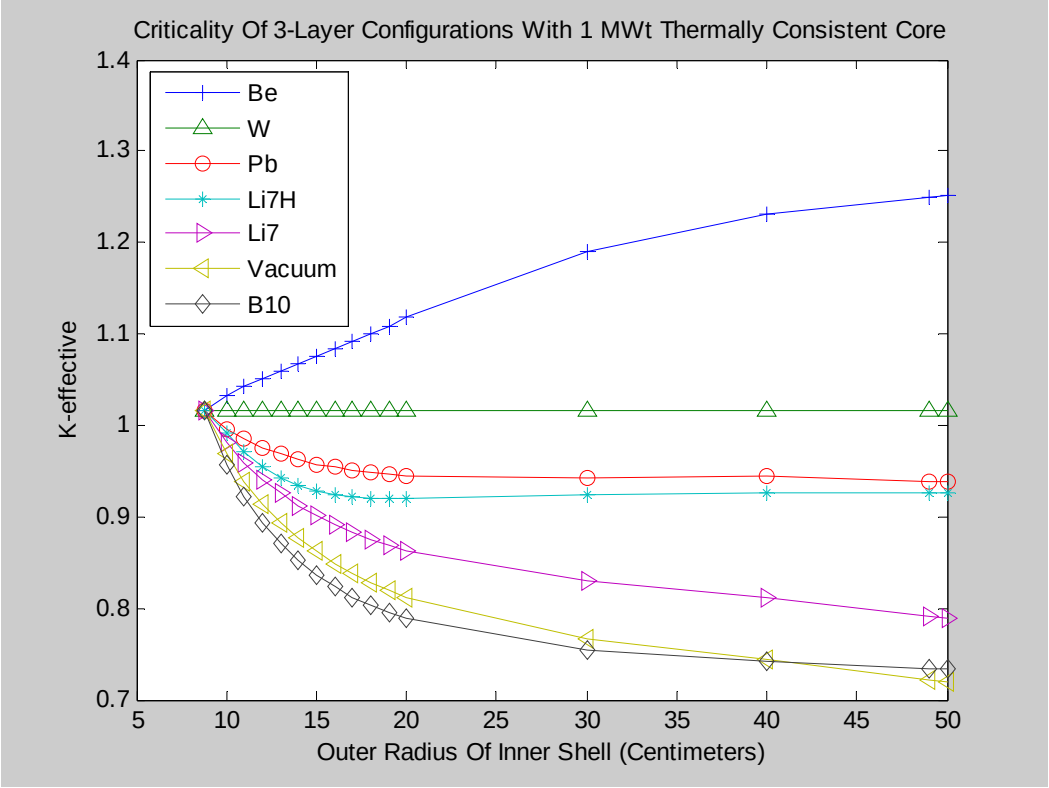


Figure 5-9: 1 MWt-Consistent 3-Layer Configuration Criticality With Different Inner Shells And With Outer Tungsten Shell To 50 cm

The per unit volumetric expansion coefficients for solid Be is -33.9E-6/Kelvin, for solid tungsten is -13.5E-6/Kelvin, for molten lithium is -230E-6/Kelvin, and for molten lead is -121.4/Kelvin. To estimate the effect of thermal expansion on reactivity it is necessary to examine the *difference* between vacuum results and the material results of Table 5-12 and multiply those differences by the thermal expansions. **The conclusion is that lead (Pb) provides about 50% more reactivity change due to thermal expansion than lithium, so use of molten lead in a shell for thermal control of reactivity seems attractive**.

A configuration is therefore envisioned in which a thick tungsten shell surrounding the reactor core includes an inner shell cavity closely surrounding the core. It would be partly filled by wedge shaped solid pieces of Be arranged to form a thin shell at the cavity's inner edge, and would also have a mechanical apparatus operated under external control to reposition these wedges farther from the core. The remainder of the cavity volume would be filled with molten lead connected to an external volume of lead so that thermal expansion and contraction would cause lead to leave or enter the cavity. This would provide a thermally coupled stable reactivity coefficient based completely on a passive mechanism. Additionally, the molten lead could be drained from the cavity in an emergency shutdown to drop reactivity to the low levels associated with a vacuum gap.

As an example, consider a configuration in which tungsten fills the spaces 8.77<R<10 cm and 20<R<50 cm but the cavity containing Be pieces and molten lead extends from R=10 to R=20 cm. The Be pieces, when positioned in their minimum radius locations, form a shell filling the volume 10<R<15 cm. When the Be pieces are positioned further out, they do not have enough volume to completely fill shells at those locations. Dividing the cavity into subshells, the controllable positions of the beryllium pieces were represented by the volume fractions listed in Table 5-13 in subsequent CSAS criticality runs: CSAS k-effective results for different molten lead densities appear in Table 5-14.

Table 5-14 shows that reactor shutdown would be assured if the Be shell is driven out while simultaneously the molten Pb is drained, since the change in reactivity worth between that condition and the opposite (i.e., Be shell fully inwards and cavity filled with Pb) is about 13%, which is quite large. This would provide a shutdown margin of 10% to 13% depending on time in the reactor core's useful life. In comparison, a typical LMFBR shutdown margin ranges from only 2.4% to 3.7% [Duderstadt and Hamilton 1976, 539].

Table 5-14 also shows that the controllable reactivity adjustment available by radially moving the Be shell pieces without also draining the molten lead is about 0.051. Since the total delayed neutron fraction for fast reactors with U235 fuel is 0.0064 [Stacey 2001, 140], it follows that this is equivalent to 0.051/0.0064= $7.97, i.e., 7.97 times the amount of reactivity needed to transition from operation at constant power to prompt critical.

The temperature coefficient of reactivity arising from the molten lead's thermal expansion can be estimated from the difference between the first two rows in Table 5-14. The resulting estimated thermal reactivity coefficients for a fixed-volume molten spherical shell reflector surrounding a fast reactor core are listed in Table 5-14.

Table 5-13: Volume Fractions For Radially Movable Beryllium Shell In Molten Lead

	ΔR_{Be}=0 cm		ΔR_{Be}=1 cm		ΔR_{Be}=2 cm		ΔR_{Be}=3 cm		ΔR_{Be}=4 cm		ΔR_{Be}=5 cm	
	Be	Pb	Be	Pb	Be	Pb	Be	Pb	Be	Pb	Be	Pb
10<R<11	1	0	0	1	0	1	0	1	0	1	0	1
11<R<12	1	0	0.8338	0.1662	0	1	0	1	0	1	0	1
12<R<13	1	0	0.8465	0.1535	0.7058	0.2942	0	1	0	1	0	1
13<R<14	1	0	0.8574	0.1426	0.7258	0.2742	0.6051	0.3949	0	1	0	1
14<R<15	1	0	0.8669	0.1331	0.7433	0.2567	0.6292	0.3708	0.5246	0.4754	0	1
15<R<16	0	1	0.8752	0.1248	0.7587	0.2413	0.6505	0.3495	0.5506	0.4494	0.4591	0.5409
16<R<17	0	1	0	1	0.7723	0.2277	0.6695	0.3305	0.5741	0.4259	0.4859	0.5141
17<R<18	0	1	0	1	0	1	0.6866	0.3134	0.5952	0.4048	0.5103	0.4897
18<R<19	0	1	0	1	0	1	0	1	0.6144	0.3856	0.5326	0.4674
19<R<20	0	1	0	1	0	1	0	1	0	1	0.5530	0.4470

Table 5-14: K-effective Values of Radially Movable Spherical Beryllium Shell In Molten Lead

Displacement of Be Shell =	ΔR_{Be}=0 cm	ΔR_{Be}=1 cm	ΔR_{Be}=2 cm	ΔR_{Be}=3 cm	ΔR_{Be}=4 cm	ΔR_{Be}=5 cm
1100 Kelvins Pb ρ_{Pb}=10.050g/cm^3	1.03449	1.01681	1.00444	0.99550	0.98952	0.98399
1200 Kelvins Pb ρ_{Pb}=9.928 g/cm^3	1.03414	1.01623	1.00370	0.99465	0.98859	0.98299
DRAINED Pb ρ_{Pb}=0 g/cm^3	1.00622	0.96794	0.94230	0.92472	0.91253	0.90402

Table 5-15: Reactivity Temperature Coefficient From Molten Lead

Displacement of Be Shell =	ΔR_{Be}=0 cm	ΔR_{Be}=1 cm	ΔR_{Be}=2 cm	ΔR_{Be}=3 cm	ΔR_{Be}=4 cm	ΔR_{Be}=5 cm
Δkeff/keff/ΔT (Kelvins^{-1}	$-3.4 * 10^{-6}$	$-5.7 * 10^{-6}$	$-7.4 * 10^{-6}$	$-8.5 * 10^{-6}$	$-9.3 * 10^{-6}$	$-10.2 * 10^{-6}$

For comparison with Table 5-15, reactivity temperature coefficients for a typical 1000 MWe oxide fueled reactor total about -12.3 * 10^{-6} [Stacey 2001, 168], but that includes contributions from fuel expansion and other stabilizing effects which have not been estimated here. Thus, it appears according to these approximate estimates that this molten lead reflector reactivity control scheme could provide stable performance in the right ballpark numerically, and it may be sufficient for this role.

CHAPTER 6
6 RADIATION PROTECTION APPROPRIATE FOR MARS

6.1 Summary

Certain types of radiation, including some of the radiations emanating from nuclear fission reactors or from cosmic or solar radiation sources in space, have the property that as the radiations pass through and interact with materials they cause some atoms to dissociate into electrons and positively charged ions. The resulting free ions and free electrons within the irradiated material quickly recombine, but frequently do so in a different chemical pattern than they had prior to ionization. The resulting random changes in the material's internal chemical structure degrade the physical properties of solid materials (such as plastics) after many ionization and recombination events have occurred with a cumulative high spatial density. However, living tissues or organs depend more sensitively on their extremely detailed internal chemical structure so are far more susceptible to damage from random internal molecular changes than are inanimate solid materials. Life processes are detrimentally affected in measurable ways by physically small doses of ionizing radiation. Radiation doses high enough to cause almost immediate death are physically so small that their deposited heat is too little to notice and the property changes in similarly irradiated inanimate solids cannot be detected.

This chapter reviews relevant radiation protection background material and discusses the approach appropriate for manned Mars missions. The purpose is to set radiation shielding requirements for the nuclear engines of the present work.

In principle, no radiation shield can completely block all radiation. Any particular shielding design will permit a design-dependant fraction of radiation to leak through it. However, radiation leakage can be kept as small as desired by designing suitably massive shields. Thus, there is an inherent design trade-off between radiation leakage and shield mass. Insisting on extremely low limits for radiation leakage would force designed shields to have very high mass, perhaps too high for practical surface vehicles on Mars. On the other hand, although high levels of permitted radiation leakage would allow a light-weight shield, it might also prove unhealthy for intended human occupancy durations.

A proper study of shielding trade-offs should consider many factors. For scientist astronauts on an expedition from Earth to Mars the existence of many potentially lethal immediate risks not related to radiation exposure should be balanced against the theoretical increase in probability that many years after successfully returning from Mars they might die early from a fatal cancer. The high radiation background doses from cosmic and solar sources which astronauts will absorb should also be considered, since it does not make sense to limit a nuclear reactor's radiation shield leakage to a tiny fraction of the natural background. Since the reactor for a mobile vehicle will not operate at full

rated engine power all the time, expected human occupancy times for different spatial zones around the reactor should be estimated along with the likely reactor power levels while people are occupying them. For example, much less shielding should be allocated to protect a suited astronaut rarely approaching the nuclear reactor engine from the side than to protect astronauts inside a continuously occupied vehicle powered by the same engine.

6.2 Radiation Dose Unit

The fundamental measurement characterizing ionizing radiation in any material such as the living tissues of the human body is the ratio of the total radiation energy absorbed to the total material mass. Early in the development of nuclear technology the centimeter-gram-second (cgs) version of the metric system was commonly employed, so its energy unit, the erg, was used along with its mass unit, the gram, for radiation dose measurements. The **R**adiation **A**bsorbed **D**ose was measured then in the combined units, $(erg)(g)^{-1}$, a combination which was assigned the special unit name, "rad". More recently the Systeme Internationale (SI) version of the metric system based on meter-kilogram-second units has replaced the cgs version in most scientific and engineering arenas. Today the SI energy unit, the joule, is used along with its mass unit, the kilogram, to measure the radiation absorbed dose in the combined units of $(joule)(kilogram)^{-1}$, a combination which in 1975 was assigned the special unit name, "Gray", in honor of the British physicist, Louis Harold Gray (1905-1965). The Gray is commonly abbreviated in text as "Gy". Because both the rad and the Gray are derived from the metric system the conversion between the two radiation absorbed dose units is simple. One rad equals one centiGray (1 cGy) equals ten milliGray (10 mGy), or, equivalently, one Gray equals 100 rads. Both units are used herein in referencing other documents but the Gray unit is preferred.

Identical absorbed doses of different types of radiation can have very different biological consequences and so cannot be directly used to predict the biological detriment to an individual resulting from radiation exposure. For instance, 0.1 Gy of neutron exposure is much more damaging to humans than 0.1 Gy of gamma rays. The "Equivalent Dose" radiation measurement was therefore invented to renormalize radiation exposure units so that identical values on a single scale would have approximately the same biological consequences. Since x-rays and gamma rays had been found to have about the same biological effects for the same radiation absorbed dose values, they were chosen as the standard. The equivalent dose of some radiation (e.g., neutrons of 10 keV energy) would be the radiation absorbed dose of gamma rays which produce an equivalent biological effect.

At this point an assumption was made that the different biological damage functions of radiation absorbed dose for the different types of ionizing radiation are *linearly* related to each other. This assumption may not be entirely justified but it has simplified radiation protection in practice and has been adopted worldwide. Therefore, the “equivalent dose"

was defined as the product of the radiation absorbed dose and a dimensionless radiation quality factor which takes on different values for different energies or types of ionizing radiation. These factors are variously termed "radiation weighting factors" or "radiation quality factors". By definition these factors are one (1) for x-ray and gamma ray radiation. For other types of radiation, appropriate radiation weighting factor values have been recommended by official expert groups such as the National Council on Radiation Protection and Measurements (NCRP) and the International Committee on Radiological Protection (ICRP), which have collectively reviewed and analyzed published experimental scientific results from radiobiological investigations. These expert groups, which have issued almost identical recommended weighting factor values, occasionally publish refined factor value updates as more experimental radiobiological scientific information becomes available. Table 6-1 reproduces the particular radiation weighting factors recommended in the NCRP's report number 116 (issued in 1993).

For radiations not listed in this table the committees also provide formulae prescriptions to estimate radiation weighting factors from Linear Energy Transfer (LET) measurements or calculations.

An issue important for space missions is that appropriate radiation weighting factors for galactic cosmic rays are not accurately known. The galactic cosmic radiation permeating deep space includes heavy particles up through uranium with particle energies ranging up to 10^{21} eV. This is well beyond the energy ranges in which radiobiological experiments have been performed, so there is not much confidence in the radiation weighting factors which have been assigned to them.

Since the weighting factors for different radiations are all dimensionless, the equivalent dose carries the same combination of fundamental energy per unit mass units as the radiation absorbed dose. Therefore, to distinguish between the *meanings* of absorbed and equivalent doses the combined units have been assigned different names. The old unit for effective dose was called the "rem", short for "rad equivalent man". The new unit is called the "Sievert" in honor of the Swedish medical physicist Rolf Maximilian Sievert (1896-1966) who contributed to radiobiology's development. The Sievert unit is commonly abbreviated as Sv. Because of their definitions the Sievert and rem can be easily converted to each other, since one rem equals one centiSievert (1 cSv) equals ten milliSievert (10 mSv) or equivalently one Sievert equals 100 rem. Both rem and Sievert units are used herein when referencing existing documents but the Sievert unit is preferred.

6.3 Typical Background Radiation Levels

The natural background of ionizing radiation on Earth's surface is low but varies widely with location. An average background rate was estimated ([Turner 1995, 11] as three milliSieverts per year (3.0 mSv y^{-1}), while medical procedures and consumer products have been estimated to add another 0.6 mSv y^{-1} to the average. The National Academy of

Table 6-1: Radiation Weighting Factors From *NCRP Report Number 116 (1993)

Radiation Type	**w_R**
X-rays, gamma-rays, electrons, positrons, muons	**1**
neutrons, energy < 10 keV	**5**
neutrons, 10 keV < energy < 100 keV	**10**
neutrons 100 keV < energy < 2 MeV	**20**
neutrons 2 MeV < energy < 20 MeV	**10**
neutrons 20 MeV < energy	**5**
protons 2 MeV < energy	**2**
alpha particles, fission fragments, nonrelativistic heavy nuclei	**20**

*** Excerpted from [Turner 1995, 434]**

Sciences has for several decades sponsored authoritative committees of experts to study the **B**iological ***E***ffects of ***I***onizing ***R***adiation. Their latest report, the BEIR VII Phase 2 report issued in 2006, estimates the average natural ionizing background radiation level worldwide as 2.4 mSv y^{-1} but varying with location on Earth's surface over the range from 1 to 10 mSv y^{-1}. The natural background arises from two sources, the decay of natural radioactive isotopes found in or released from Earth's crust, such as radon gas, and the radiation penetrating Earth's atmosphere originating from cosmic or solar sources. Earth's atmosphere is an effective radiation shield with a mass of about 10 tonnes above each square meter of Earth's surface, i.e., about 1,000 g/cm^2. A similarly massive shielding layer using liquid water would be 10 meters (i.e., 33 feet) deep.

Background ionizing radiation in space is much more intense than on Earth's surface and is also much more complicated. Most ionizing radiation in space occurs in the form of high speed charged particles rather than neutral particles. It arises from two distinct sources, a distant Galactic Cosmic Ray (GCR) background isotropic source from outside the solar system, and Solar Proton Events (SPEs) in which during relatively brief solar flares or Coronal Mass Ejections (CMEs) the sun emits intense fluxes of high speed ionized hydrogen plasma. Typical SPE durations range from one to several days. The GCR background consists of extremely high energy particles including heavy atomic nuclei with individual kinetic energies extending up to 10^{21} eV. SPE radiation consists of protons with individual energies typically around 10^7 eV but having an energy spectrum extending up to about 10^8 eV. SPE protons are strongly affected by magnetic fields which have a size scale as small as Earth's magnetic field, so many of them are trapped in the Van Allen radiation belts surrounding the Earth. Thus, Earth's magnetic field reduces radiation doses in low Earth orbit locations such as the orbit of the International Space Station. There is no similarly strong magnetic field to deflect SPE protons in either interplanetary space between Earth and Mars or near Mars, which lacks a planetary magnetic field at the present time. The much higher energy GCR radiations are not so strongly affected by Earth's magnetic field but are repelled by the solar-system-scale magnetic fields carried by the solar wind plasma as it streams away from the sun. Since the solar wind varies in intensity in step with the 11 year solar sunspot cycle, the GCR background radiation penetrating to the inner solar system near and between Earth and Mars also varies but in opposite phase, with the total intensity of GCR radiation varying by about a factor of two over the cycle.

In low Earth orbit (LEO) at the International Space Station (ISS), the *unshielded* average background radiation ranges from about 160 to 320 mSv y^{-1}, depending on the time in the solar sunspot cycle. (Typical astronaut doses are in the 50 to 100 mSv range due to limited stay times in LEO aboard the ISS and to the shielding provided by its walls). The MARIE experiment on the Mars Odyssey spacecraft found the *unshielded* radiation level in low Mars orbit (LMO) was a stated 2.5 times as high as at the ISS, so it follows that the background radiation in LMO was about 400 to 800 mSv y^{-1}, depending again on the time in the solar cycle. In LMO the planet Mars must be filling almost half of the 4π steradian field of view visible from a spacecraft, so since Mars lacks a magnetic field the planet itself must have been shielding almost half of the radiation background that would

be felt in interplanetary space far from either Earth or Mars. That observation leads to an estimate that average total *unshielded* background radiation during interplanetary transfers may vary between 800 and 1600 mSv y^{-1}, depending on solar activity.

Astronauts will never be subjected to the full unshielded background radiation level in space because they will always be surrounded by a pressure-tight boundary. Even a spacesuit provides a minimal level of shielding but most spacecraft have somewhat thicker walls and so provide more shielding. Larger spacecraft like the ISS also have small regions with additional shielding installed for astronauts to retreat into during solar flare radiation events. Even thicker radiation shielding could in principle be provided in spacecraft but during the present century spacecraft will likely remain too small to include a radiation shield approaching the 1,000 g/cm^2 integral-mass-density of Earth's atmosphere.

As stated herein in Chapter 2, relatively brief Solar Proton Event (SPE) bursts account for much of the space radiation beyond LEO. As measured by the MARIE experiment on the 2001 Mars Odyssey orbiter spacecraft in low Mars orbit before the MARIE equipment failed, the GCR radiation component at that time was only about 80 mGy y^{-1}. On the other hand, the planet itself was shielding some of the GCR so the interplanetary GCR background must have been higher, perhaps 160 mGy y^{-1}. Using measurement data acquired during the 1970s, the unshielded GCR background during interplanetary transfers between Earth and Mars has been estimated as 179 mSv y^{-1} during solar maximum and 488 mSv y^{-1} during solar minimum. Because of the very high GCR particle energy it would be difficult to significantly reduce this in a small spacecraft through the shielding provided by its conventional walls. Calculations have estimated that 10 g/cm^2 of aluminum shielding (corresponding to a 3.7 cm thickness) would only reduce the GCR dose to 154 mSv y^{-1} during solar maximum and to 393 mSv y^{-1} during solar minimum [Townsend 2005]

As opposed to the relatively constant GCR background, the SPE dose is highly variable and no accurate method exists at present to predict it in advance. Individual brief SPEs caused by solar flares or CMEs can pose a radiation hazard. In NASA Technical Paper 3137 (1993), M. Weyland et al analyzed the largest solar flare SPEs that occurred between 1950 and 1989, a period during which scientific observations provided well documented records suitable as starting points for calculations. The SPEs analyzed included one in February 1956, one in November 1960, one in August 1972, and three respectively in August, September, and October of 1989 which were combined since their spacing in time was small enough to affect a single mission to Mars. This study found that even behind a 5 g/cm^2 layer of aluminum shielding (i.e., a layer about 1.85 cm thick), the radiation doses in interplanetary space from the 1972 and 1989 SPEs would have each exceeded 500 mSv. More rarely, SPEs may involve even more significant radiation doses, although there is little historically recorded measurement data from the random distribution of actual past SPEs to characterize them. However, SPE events influence isotope production in Earth's atmosphere which in turn leaves a record in the annual snowfall deposits accumulating in Earth's glaciers. The Carrington Solar Flare of 1859

has recently been identified based on ice core data from Antarctica as being the largest such SPE affecting Earth in the past 500 years. Data about it are sparse but reconstructions [Townsend 2005]estimate its interplanetary unshielded dose as having been somewhere in the range from 1000 mGy to 3000 mGy, and also estimate that a radiation shielding thickness equivalent to 18 cm aluminum would be needed to reduce its radiation dose to 250 mGy.

6.4 Acute Effects of Radiation

Biological repair mechanisms exist to counter some of the detrimental effects of radiation doses received slowly over extended time durations, assumed herein to be durations longer than 30 days. Herein any radiation dose received over a period of 30 days or less is defined to be an *acute* dose. If a large whole-body radiation dose is received suddenly over a short enough time period there is not sufficient time for the biological repair mechanisms to act, as vital tissues and organs are damaged simultaneously. Large acute doses of radiation can produce *acute radiation syndrome*, which must certainly be avoided in manned Mars missions.

Relatively small whole-body acute gamma ray doses less than 0.25 Gy produce no clinically significant effects and larger acute doses up to 1 Gy produce mostly no symptoms with a small fraction of exposed people exhibiting temporary nausea and anorexia. At the 1 Gy acute exposure level there are decreases in blood cell and platelet counts and damage to bone marrow, lymph nodes and spleen, but essentially all exposed people recover. Fatalities will occur in the 3 to 5 Gy range without treatment, with 4 Gy probably fatal within 30 days to 50% of the people exposed. Acute doses of 6 Gy or more kill essentially everyone [Turner 1995, 403].

No manned Mars missions will plan for astronauts to receive large acute radiation doses. If such an acute dose were received due to an accident or mistake, its detrimental biological effects would likely be less survivable than similar acute radiation exposures occurring on or close to Earth where there is ample help available. (For instance, consider radiation-induced nausea inside a spacesuit.) Therefore it is prudent to try to ensure that no dose in any 30 day period ever exceeds 250 mSv and to classify 1,000 mSv or larger acute doses as being similar to other non-survivable scenarios in which crew members most likely perish, such as tearing a spacesuit during a walk outside or crashing a rocket vehicle on Mars.

6.5 Stochastic Effects Of Chronic Low-Dose Radiation

The primary detrimental effect of chronic radiation at dose rates too low to cause acute radiation syndrome is an increased risk of developing cancer at a later time, perhaps decades after the exposures occurred. This effect is well known and is well established as being quite real but due to its stochastic nature and the paucity of data relating low dose

radiation exposures to cancer incidence, existing predictive models for it have an inherent high level of uncertainty. Studies of survivors of the Japanese atomic bomb survivors provide most of the data used to estimate this dose-response relation. That study has some inherent problems, including the fact that no individual radiation dose measurements were ever made. Instead, only estimates of individual radiation doses were made based on remembered locations and movements of the individuals at the time of the blasts and during the days and weeks following, and also based on computer simulations estimating what the likely radiation doses were in different times and regions. Also, the atomic bomb survivors' radiation doses were acute, not chronic.

The most recent dose-response relation developed to predict cancer incidence and mortality attributed to low dose radiation exposures was documented and discussed in Chapter 12 of the National Academy of Science's BEIR VII Phase 2 report (2006). For fatal cancers, their Exposure Additive Risk "EAR" model as presented in the report takes the following form:

$$\lambda(c,s,a,b,d) = \lambda'(c,s,a,b) + \beta_s \exp(\gamma e^*) a^\eta d$$

where (6-1)

$$e^* \equiv \begin{cases} e - 30 & \text{if e} < 30 \\ 0 & \text{if e} \geq 30 \end{cases}$$

In Eq. 6-1, d is the dose in Sv while λ is the predicted fraction of the population that die due to cancer, c is the city the exposed person lives in during and following the exposure, s is the person's sex, a is the person's attained age in years for which the cancer probability evaluation is being made, b is birth cohort of the person, and e is the person's age at the time the radiation exposure occurred. The report provides recommended values for the parameters γ and η but omits describing the dependence on city and birth cohort, instead providing recommended values to use for the λ' term for Americans of either sex regardless of their attained age. Recommended EAR model values for cancer mortality were as in the Table 6-2. The listed confidence intervals are sufficiently large that mortality prediction *rates* using this formula are quite uncertain, but this is the best available.

This formula and its parameters predict that females are slightly more susceptible to radiation induced cancer mortality than males. More importantly, it also predicts that young people less than 30 years old are considerably more susceptible than people over 30, and that the younger the person is under 30 the more danger a given dose of radiation will pose. Although Eq. 6-1 does not predict a continued decrease in radiation effects with increasing age for exposed people over 30, it does predict that the longer their time living after exposure the greater their chance of developing a fatal cancer as a result of the exposure. Therefore, people that are much older than 30 at the time of exposure have a smaller chance of dying from a radiation induced fatal cancer than 30 year olds since there is more chance that the older people will succumb to some other disease or old age

Table 6-2: BEIR VII Recommended Values For Cancer Mortality Prediction

Parameter	Recommended Value	95%Subjective Confidence Interval
λ'	**0.10127**	
β_{male}	**1.1e-04**	**(0.75e-4, 1.7e-4)**
β_{female}	**1.3e-04**	**(0.98e-4, 1.8e-4)**
γ	**-0.037**	**(-0.059, -0.015)**
η	**3.5**	**(2.71, 4.18)**

before a radiation induced cancer could kill them. For these reasons and others it seems likely that the crews sent to Mars during the present century will not include people younger than 30, and will likely be made up of more seasoned scientists and engineers in their 40s, 50s and perhaps even 60s.

While Eq. 6-1 predicts a probability of cancer mortality, it does not by itself estimate the Lifetime Attributable Risk (LAR) of dying from cancer as a result of radiation exposure. To predict the LAR requires that competing causes of mortality also be modeled in the analysis, which as a result becomes complicated. For instance, a decrease in heart disease mortality unrelated to radiation would cause the LAR cancer mortality rate attributable to radiation to increase, simply because of its definition. The 2006 BEIR VII report does provides some calculated values for this quantity on its page 281; they are reproduced here as Table 6-3.

Table 6-3 predicts that a 1 Sv= 1000 mSv exposure received over the almost 3 year duration of a manned Mars mission would increase the chance of a crew member eventually dying from cancer long after the mission has finished by about 3.2% if the person is a 40 year old male, by about 2.9 % if the person is a 50 year old male, by about 4.9 % if the person is a 40 year old female, or by about 4.2 % if the person is a 50 year old female. The large confidence intervals show that these values are not much better than order of magnitude estimates. However, even if they were known with certainty these resulting mortality increments would be difficult or impossible to statistically discern in the small population of people expected to actually visit Mars during this century. Since the present chance of eventually dying from cancer without any additional radiation exposure is now about 42% in the general USA population, these are not remarkable mortality rate changes.

Since manned missions to Mars will probably begin more than 15 years from now, it is possible that the situation concerning cancer mortality risk may have changed dramatically by then. (1) If an effective medical cure for all cancers is developed then these cancer induction risks would no longer be of concern. (2) If the present newly available medical diagnostic tools for early detection and treatment of cancers are extended and fully deployed for returning crew members, a larger fraction of their cancers would be cured so their mortality risk from radiation exposure would be reduced. (3) Recent research suggests some people are more susceptible to radiation induced cancers than others due to specific inherited genetic defects. It may be possible to develop a genetic screening approach so that only astronauts who are more resistant to radiation would be chosen for Mars missions, thus reducing the cancer induction risk.

6.6 Present Recommended Radiation Exposure Limits for Earth

The accepted approach to terrestrial radiation protection ignores the background dose level and also does not directly control radiation doses resulting from medical procedures or diagnostic instruments. Instead it limits the *additional* radiation exposures which are

Table 6-3: Lifetime Attributable Risk: Cancer Deaths Per 100,000 Attributable To Radiation

Exposure Scenario (Low LET Radiation, e.g., Gamma Rays)	**Males Deaths (95 % ConfidenceInterval)**	**Females Deaths (95 % ConfidenceInterval)**
0.1 Gy to population of mixed ages	**410 (200, 830)**	**600 (310, 1230)**
0.1 Gy at age 30	**640 (300, 1390)**	**1050 (470, 2330)**
0.1 Gy at age 40	**320 (150, 650)**	**490 (259, 950)**
0.1 Gy at age 50	**290 (149, 600_**	**420 (210, 810)**
1 mGy per year throughout life	**290 (140, 580)**	**460 (230, 920)**
10 mGy per year from ages 18 through 65	**1410 (700, 2860)**	**2170 (1130, 4200)**

under the control of a responsible agent, classifying them as either *occupational exposure* or *exposure of the public*. Its primary purpose is to limit cancer induction due to radiation exposure. Basic exposure limits as recommended by the NCRP and the ICRP are summarized in Table 6-4 [Turner 1995, 441]. The limit on exposures to the public creates a situation in which the additional radiation exposures due to artificial radiation sources are even smaller than the differences between background radiation levels at different locations where people live. Since members of the public are typically unconcerned with health effects of background radiation levels varying between different geographical locations, this limit helps assure the public about the safety of radiation sources. However, such limits on exposures to the public are utterly irrelevant for Mars.

The cumulative occupational exposure limit of Table 6-4 corresponds to the last line of Table 6-3. Together they assert that the Lifetime Attributable Risk of cancer for a person who receives annual doses of 10 mSv y^{-1} every year from age 18 through age 65 (thus keeping up with the recommended maximum cumulative occupational exposure limit vs. age) would only be 1.4% for males and 2.1% for females. These are such low rates that it would be difficult to statistically discern them from mortality studies of retired radiation workers. Indeed, the Naval Shipyard study found lower cancer rates, a fact suggesting (as stated by the American Nuclear Society) that the BEIR reports based on Japanese atom bomb survivors may *overstate* the dangers of chronic low dose radiation.

The Table 6-4 recommended occupational exposure limits, like the limits on exposures to the public, appear to have been chosen to make the resulting health risk per person small enough that few would object. This is appropriate for terrestrial situations where the background radiation level is very low and where for the benefit of an employer a radiation dose is accepted by an employee who has no personal motive beyond a paycheck to accept the resulting health risk. It does not appear appropriate for astronauts on missions to Mars during the present century who will face high *background* levels of radiation and who, to a very great extent, **will be self-selected volunteers choosing to take substantial non-radiological risks to their lives in the service of science**.

6.7 USA Legal/Administrative Radiation Dose Limits For Space

Although USA legal and administrative details are more subject to change over time than real physical details and their applicability may be limited to only astronauts from the USA, the USA's radiation dose limits for manned space missions are important constraints which are enforced at the present time. It is possible that they may still remain in effect without change when manned missions to Mars by the USA begin 15 or more years from now, so therefore they are discussed here.

Table 6-4: Present Recommended Basic Radiation Protection Exposure Limits For Earth

Recommending Document:	**NCRP-116**	**ICRP-60**
Occupational Exposure: Annual **Cumulative**	**50 mSv y^{-1}** **10 mSv * age(y)**	**50 mSv y^{-1}** **100 mSv in 5 y**
Exposure to the Public	**1 mSv y^{-1} if continuous**	**1 mSv y^{-1} for 5 y average**

NASA maintains a Spaceflight Radiation Health Program at its Johnson Spaceflight Center, whose Technical Memo TM104782 summarizes the present radiation protection for manned spaceflight. It states that US astronauts are officially classified as radiation workers but that this classification does not subject them to the radiation exposure occupational limits applied to terrestrial workers in the domestic nuclear power industry. Instead, this classification simply requires a formal program must exist to protect them from excessive exposure. Presidential Executive Order 12196 requires federal agencies including NASA to comply with Occupational Safety and Health Administration (OSHA) regulations for ionizing radiation exposure, but no OSHA standards exist for spaceflight so they have been judged inappropriate. This document goes on to state:

> "Terrestrial radiation exposure guidelines provided in the Code of Federal Regulations (29CFR1910.96) are too restrictive for space activities and have been judged to be inappropriate. NASA can establish supplementary standards for appropriate control of radiation for astronauts in accordance with 29CFR1960.18. The following NASA requirements serve as a basis for the implementation of the supplementary standard:
>
> (1) that its use applies to a limited population,
> (2) maintenance of detailed flight crew exposure records,
> (3) preflight hazard assessment/appraisal,
> (4) planned exposures be kept As Low As Reasonably Achievable (ALARA),
> (5) maintenance of operational procedures and flight rules to minimize the chance of excessive exposure, and
> (6) man-made onboard radiation exposure complies with 29CFR1910.96 except where the NASA mission objectives cannot be accomplished otherwise."

It is important to note NASA requirement number (6) listed above administratively requires radiation from the mobile nuclear engines of the present work to be treated differently from the intense external background radiation which may be received from solar or galactic sources. Although this may seem not entirely rational it is the present rule and is discussed further below. The document goes on to state:

> "NASA has adopted the recommendations that the National Council On Radiation Protection and Measurement (NCRP) presented in its Report 98, "Guidance On Radiation Received in Space Activities" (July 1989) as the basis for the supplementary standard for spaceflight crew radiation exposures. The maximum exposure limits are presented in Tables 1 and 2. Whereas monthly and annual limits exist to prevent the short term physiological effects of exposure, cancer limits exist to contain radiation risk within a 3% increased lifetime cancer mortality. The recommendations of the NCRP apply to activities in low Earth orbit, such as Space Station. Astronaut exposure limits are greater than those of terrestrial radiation workers."

TABLE 1. ORGAN SPECIFIC EXPOSURE LIMITS

EXPOSURE INTERVAL	DEPTH (5 CM)	EYE (0.3 CM)	SKIN (0.01 CM)
30 DAYS	25 REM	100 REM	150 REM
ANNUAL	50 REM	200	300
CAREER	100 TO 400	400	600

TABLE 2. CURRENT CAREER EXPOSUURE LIMITS BY AGE AND SEX*

SEX	AGE			
	25	35	45	55
MALE	150 REM	250 REM	325 REM	400 REM
FEMALE	100	175	250	300

The career depth equivalent dose limit is based upon a maximum 3% lifetime career risk of cancer mortality. The total equivalent dose yielding this risk depends on sex and age at the start of exposure. The career equivalent dose limit is approximately equal to:

200+7.5(age-30) rem for males up to 400 rem maximum

200+7.5(age-38) rem for females up to 400 rem maximum."

The above Tables 1 and 2 with their associated text as excerpted from NASA's documents apply to total radiation received in space missions from environmental and manmade radiation sources together. However, NASA's requirement (6) places additional constraints on radiation from manmade sources. The 29CFR1910.1096 reference that it invokes in turn contains the following additional passage:

"(b) *Exposure of individuals to radiation in restricted areas.*

(1) Except as provided in paragraph (b)(2) of this section, no employer shall possess, use, or transfer sources of ionizing radiation in such a manner as to cause any individual in a restricted area to receive in any period of one calendar quarter from sources in the employer's possession or control a dose in excess of the limits specified in Table G18. (2) An employer may permit an individual in a restricted area to receive doses to the whole body greater than those permitted under subparagraph (1) of this paragraph, so long as

(i) During any calendar quarter the dose to the whole body shall not exceed 3 rems,

(ii) The dose to the whole body, when added to the accumulated occupational dose to the whole body, shall not exceed 5(N-18) rems where "N" equals the individuals age in years at his last birthday; and

(iii) The employer maintains adequate past and current exposure records which show that the addition of such a dose will not cause the individual to exceed the amount authorized in this subparagraph. As used in this sub-paragraph, *dose to the whole body* shall be deemed to include any dose to the whole body, gonad, active blood forming organs, head and trunk, or lens of the eye.

(3) No employer shall permit any employee who is under 18 years of age to receive in any period of one calendar quarter a dose in excess of 10 percent of the limits specified in Table G18."

TABLE G18

	Rems per calendar quarter
Whole body; Head and trunk; active blood forming organs; lens of eyes; or gonads	1.25
Hands and forearms; feet and ankles	18.75
Skin of whole body	7.5

These 29CFR1910.1096 excerpted passages do not appear to have been written with space missions in mind. The limits as stated in its excerpted Table G18 seem far more restrictive than the values in Tables 1 and 2 exerpted from NASA's document. They also leave some issues open to interpretation, e.g., whether the term *occupational dose* excludes or includes the intense solar and galactic environmental background radiation to which astronauts unavoidably will be exposed during a Mars mission. It seems reasonable to exclude background radiation in the terrestrial radiation worker case where the radiation dose from manmade sources approximates the total dose. It may not be appropriate for space extraterrestrial situations where the background dose is so high.

Assuming the legal and administrative definition of *occupational dose* follows past practice and excludes environmental background radiation, then the main effect of these legal and administrative rules would be to separately limit astronauts' exposure to the radiation created by *manmade* radiation sources such as the nuclear engines of the present work. The total radiation dose from such manmade sources, ignoring the background dose, would then be restricted by 29CFR1910.1096 to not exceed either 12.5 mSv per calendar quarter or the higher maximum of 30 mSv per calendar quarter if certain restrictions on past accumulated lifetime exposures to *manmade* radiation are also met. For instance, this interpretation asserts that, independent of background radiation, a 45 year old astronaut would be allowed to receive a maximum total *manmade radiation* dose of 210 mSv radiation from nuclear reactor sources during a 21 month (i.e., 7 calendar quarters) stay on Mars' surface, provided that his cumulative lifetime occupational exposure to *manmade radiation* prior to the mission did not exceed

(5(45-18)-(7)(3))/100=1.14 Sv = 1140 mSv =114 rem.

For terrestrial applications the ALARA doctrine of the above excerpted NASA rule number (5) typically requires that reasonable amounts of money and effort be spent to reduce occupational radiation doses below the stated occupational limits. ALARA requires that engineering judgment be used to balance costs versus risks in order to decide what is reasonable. However, for manned Mars missions it does not seem reasonable to spend large amounts of money to reduce the manmade component of radiation far below the level of the intense background radiation component. Instead, reasonable efforts should try if feasible to reduce the *total* risk from their *sum*, the background radiation *plus* the radiation from manmade sources. The optimal allocation of resources to do this would try to match the marginal cost of eliminating each milliSievert of background radiation against the marginal cost of eliminating each milliSievert of radiation from manmade sources. These optimal allocation decisions should depend on detailed analyses of the costs of alternative mission designs. Such cost analyses have not been done at present so it is not now clear what the resource allocation should be for ALARA.

However, with the total background radiation exposure dose during a Mars mission expected to be in the range of 1000 to 2000 mSv (i.e., 100 rem to 200 rem), a detailed cost/benefit ALARA analysis would probably conclude that it does not make sense to reduce the component of radiation arising from nuclear reactor sources much below about

10% to 20% of that total. Thus, a reasonable ALARA goal for the *manmade component* of radiation received during an entire Mars mission would likely be in the range from 100 to 400 mSv (i.e., 10 rem to 40 rem).

6.8 Assumed Mars Limit For Exposure To Manmade Radiation

Although it does not arise from a detailed ALARA analysis, the 29CFR1910.1096 interpretation permitting a 210 mSv dose from nuclear reactor sources during a 21 month surface stay on Mars may fortuitously be a convenient dose limit to adopt. Since this limit seems to have the appropriate order of magnitude and also has some administrative or legal standing, it is adopted here. The typical radiation dose from nuclear engines of the present work should thus average less than 30 mSv (i.e., 3 rem) per calendar quarter, which is equivalent to an ***average*** **dose of 0.33 mSv per day or 13.75 microSievert per hour (13.75 μSv h^{-1}), or in terms of the older units, 1.375 millirem per hour**.

6.9 Reactor Shielding Requirements

The most restrictive shielding requirements are for areas near a nuclear engine which may be continuously occupied for long time durations during high engine power operations. Astronauts will be continuously inside a pressurized rover vehicle powered by a nuclear engine throughout surface excursions lasting weeks or even months in order to travel to locations thousands of kilometers from the Mars base. The average radiation dose from the nuclear engine they receive within the pressurized vehicle should not exceed 0.33/24=0.01375 mSv $hour^{-1}$ (=1.375 millirem/hr) with reactor power at its average level during the excursion. Since the average engine power may not be accurately anticipated, a conservative approach would instead assume the engine always operates at its rated maximum power.

It is expected that people in a pressurized vehicle powered by a nuclear engine will move freely within the cabin. By placing the nuclear engine behind the pressurized vehicle and locating cargo and life support supplies and equipment between the reactor and the pressurized cabin, passengers will benefit from the resulting additional shielding and distance.

Figure 6-1 depicts in plan view the possible layout dimensions of a pressurized Mars surface vehicle 12 m (i.e., 40') long and 4 m (i.e., 13') wide. In this configuration the pressurized vehicle and its attached nuclear engine are separate autonomous vehicles which are connected to each other through a flexible power-transmitting coupling. The two vehicles travel together as a single tandem unit via integrated servo control loops. With this layout, vehicle passengers would never approach closer than 6 m (i.e., 20') from the reactor's center, and would usually be in the front half of the vehicle between 11 m and 17 m from the reactor's center. Also, between the people and the reactor would be additional radiation absorbing objects such as tanks of fresh water, waste water, water

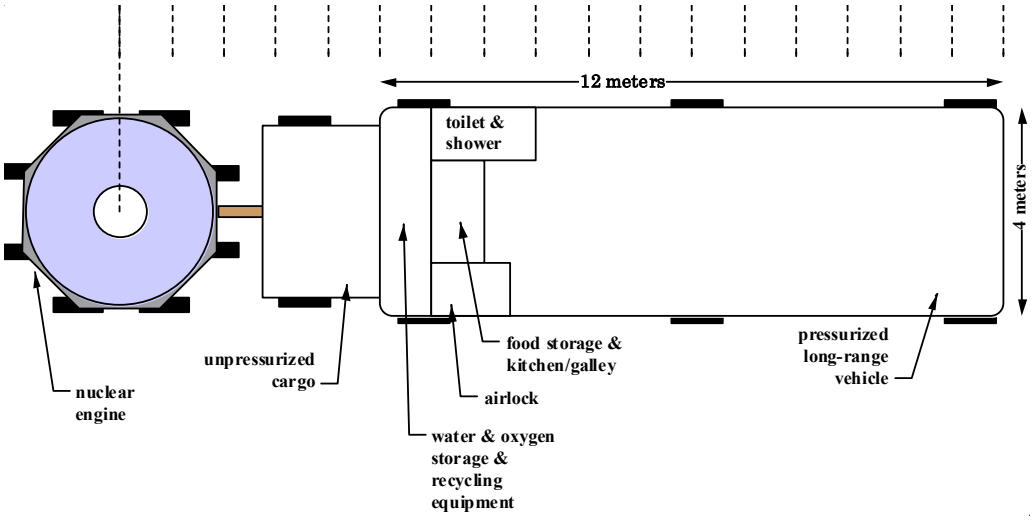


Figure 6-1: Plan View Of Mars Vehicle Powered By Nuclear Engine

recycling equipment, fresh oxygen, air recycling equipment, food supplies, communications electronics, etc. While it is not the purpose of this work to develop detailed designs for pressurized Mars vehicles, the point of this layout is the claim that necessary supplies, recycling life support and other equipment essential for long distance excursions on Mars surface could provide additional shielding and separation from a nuclear engine beyond that needed for brief short distance trips.

A second shielding requirement is to protect people who are not in the pressurized vehicle powered by the nuclear engine. In most such cases those people will be engaged in Extra-Vehicular Activities (EVAs), wearing spacesuits in the neighborhood of the pressurized vehicle, either walking or driving a small unpressurized battery-operated vehicle. The fraction of the time that EVAs will be in progress on Mars' surface near a nuclear engine is not known. During the Apollo lunar missions, surface EVAs were a large fraction of the brief time that astronauts were present on the lunar surface, perhaps near 50%. On the other hand, EVAs at the ISS represent considerably less than 1% of its occupancy time. The difference between them relates to the reasons for going outside in EVAs and the alternatives. On Luna there was no way for Apollo astronauts to explore or take surface samples without donning spacesuits and going outside, so EVAs were conducted for scientific and exploration purposes and there were no alternatives. Even excursions in the small powered lunar rovers used in the last Apollo missions were EVAs. At the ISS there is no planetary surface to investigate, so EVAs there are not motivated by science or exploration goals. The only reasons for an EVA at the ISS are to install, move, or service equipment. Special grappling arms with manipulators remotely operated from inside the ISS provide alternatives for the simplest such maintenance tasks without conducting an EVA.

On Mars' surface there will be plenty of items of interest outdoors to study, but a pressurized powered vehicle equipped with modern state-of-the-art external manipulators or even autonomous robotic devices operated remotely from inside the vehicle will allow

much of that study to proceed without putting on a spacesuit. However, EVAs will certainly occur on Mars since they are superior in accomplishing some tasks. An EVA can provide the best way to either observe something not easily monitored by remote means or to interact with equipment or with the martian surface environment in a highly flexible way that cannot be accurately anticipated in advance. Because there will be alternatives to EVAs for many purposes it is expected that EVAs will occur only a small fraction of the time but more often than on the ISS. It is therefore assumed here that the EVA time fraction for each crew member will not exceed 10%, i.e., 2.4 EVA hours/day or 72 EVA hours/month.

To estimate the radiation dose from the nuclear engine to astronauts during EVAs it is appropriate to use joint distributions of expected reactor power and occupancy of nearby locations. Since this is not known some plausible assumptions will be made. It is assumed that occupancy in physical contact with the nuclear engine is excluded while the reactor is operating at full power, and that astronauts will never crawl under it, climb on top of it, or otherwise approach it closer than 2 meters from its center while it operates at any significant power level.

The astronaut EVA occupancy distribution model assumed here is that while the nuclear engine operates continuously at its maximum rated power, astronauts during their EVAs may randomly visit with constant areal probability density all surface locations within the annular region centered on the nuclear reactor's center and extending between 2 meter and 200 meter radius circles. In this model, while the reactor is operating at full rated power astronauts would spend no more than about 0.01% of their EVA time (i.e., 26 seconds per month) close enough to the reactor to touch the engine vehicle (i.e., between 2 and 3 meters from the reactor's center). While the reactor is operating at full power astronauts would spend no more than about 1% of their EVA time (i.e. 43 minutes per month) within 20 meters from the reactor's center. Thus, 99% of the EVA time is assumed to be well away from the pressurized vehicle and its engine. Indeed, this model predicts that 75% of EVA time would be spent more than 100 meters away, and 50% of the EVA time would be more than 140 meters away.

It is understood that these occupancy assumptions may be too restrictive for some activities such as those focusing on maintenance of the vehicle itself. If it becomes necessary to spend more EVA time in the immediate vicinity of the nuclear engine, the fission power level may be lowered or even stopped altogether. In addition, tele-robotic equipment may provide an ability to inspect and adjust nuclear engine components at length without an EVA while the engine operates at full power.

An elementary integration shows this assumed occupancy distribution has the same average value of r^{-2} as the following effective "average" distance from the reactor's center:

$$\frac{\sqrt{\left(\frac{r_o}{r_i}\right)^2 - 1}}{\ln\left(\frac{r_o}{r_i}\right)} = \sqrt{\left(\frac{200}{2}\right)^2 - 1} \Big/ \ln\left(\frac{200}{2}\right) = 21.7 \text{ meters}.$$

Thus, if radiation leakage from the shielded reactor behaved like a point radiation source obeying an inverse square dependence of radiation intensity on distance, then an astronaut standing 21.7 meters from the reactor's center would receive the same dose as an astronaut roaming according to this distribution.

One conservative approach to shielding requirements would require that radiation shielding limit the average dose from the nuclear engine during its full power operation to 0.01375 mSv h^{-1}, regardless of whether the astronauts are inside the pressurized vehicle or outside in an EVA. Then the fraction of time spent in EVAs would not matter; astronauts could spend 90% of their time outside if they chose to. If shielding were designed to limit the full rated power nuclear engine dose to an astronaut in an EVA 21.7 meters distant from the reactor's center, to 0.01375 mSv h^{-1}, then the dose to the same astronaut when standing beside the nuclear engine 2 meters from its center would be approximately 1.62 mSv h^{-1}, (i.e., 162 millirem/hour), assuming an inverse square law intensity dependence. This would be low enough so as to eliminate any possibility of an accidental exposure to a hazardous acute dose, since with this shielding design approach an astronaut would need to stand more than 150 hours adjacent to the nuclear engine (i.e., 2 m from its center) while it operates at full power to accumulate 250 mSv (i.e., 25 rem).

Somewhat more radiation shielding would be required between the nuclear reactor and the people within the pressurized cabin, since they would all be closer than 21.7 meters from the reactor center. Indeed, in the Figure 6-1 diagram, people in the cabin could move between 6 meters and 17 meters distance from the reactor's center. Assuming again the inverse square approximation for radiation intensity, assuming spherically symmetric shielding and assuming a dose rate of 0.01375 mSv h^{-1} at r=21.7 m when the engine is operating at its full rated maximum power, the dose outdoors 6 meters from the reactor's center would then be 0.180 mSv h^{-1} (=18.0mrem/hr) which is about 13 times the 0.01375 mSv h^{-1} target for the average dose inside the vehicle's cabin.

The most conservative average dose rate estimate for inside the vehicle’s cabin would assume that at least one of the astronauts continuously occupies the vehicle’s rear, i.e., 6 meters from the reactor’s center, and that the engine operates continuously at its rated power. Actually the doserate to an astronaut near the front of the vehicle would be reduced by an order of magnitude below the doserate near the rear, and another order of magnitude reduction in doserate woule result while the engine idles below 10% of its design power rating. However, this most conservative approach leads to the Table 6-5 doserate restrictions.

Table 6-5: Conservative Radiation Dose-Rate Limits for Full Power Engine Operation

In Vehicle 6 meters from Reactor Center	**1.375 mrem/hr**
Outdoors 6 meters from Reactor Center	**18.000 mrem/hr**

A less conservative approach to shielding requirements would take credit for the astronauts limiting their EVA time to no more than 10%. Then the allowable radiation dose rate during EVAs could be increased without increasing the overall dose by providing more shielding for the vehicle. For instance, reducing dose while in the vehicle from 0.01375 mSv h^{-1} to 0.00775 mSv h^{-1} would allow the average EVA dose rate to be increased from 0.01375 mSv h^{-1} to 0.06774 mSv h^{-1}, since

$$(0.00775 \text{ mSv } h^{-1})(0.90)+(0.06774 \text{ mSv } h^{-1},)(0.10)=0.01375 \text{ mSv } h^{-1}$$

Assuming yet again an approximate inverse square intensity dependence, the dose rate 2 meters from the reactor's center would be 7.975 mSv h^{-1} (=797.5 mrem/hr). This is enough shielding so that hazardous acute radiation exposures cannot occur quickly. Indeed, an astronaut could spend 100 minutes two meters from the reactor's center while it operates at full power, then spend the next three months of full power operation in the rear of the pressurized vehicle's cabin and still not exceed the 30 mSv (=3rem) per calendar quarter limit on radiation from manmade sources deriving from the interpretation of 29CFR1910.1096. With this amount of reactor shielding the dose outdoors at 6 meters distance from the reactor center would be 0.886 mSv h^{-1}=88.6 mrem/hr. To reduce the outdoors doserate to the 0.00775 mSv h^{-1} doserate inside the vehicle cabin would require a further attenuation factor of 0.886/0.00775/=114. This might be provided by the shielding effects of supplies and equipment at the vehicle's rear or by further distance attenuation to more typical passenger locations within the vehicle's cabin. Assuming this approach is taken to setting radiation limits, Table 6-5 would be replaced by Table 6-6.

It is also acceptable to use considerably less radiation shielding on the bottom and top of the nuclear engine where no people will ever be located during powered operations. It is assumed for the purpose of radiation dose estimation that astronauts on EVAs will always be located between 0 and 3 m above the ground surface and that the slope (i.e., the grade) of the ground surface will not exceed 10% within 200 meters of the vehicle. It logically follows that reactor radiation emitted in directions sufficiently close to the vertical will only contribute to radiation exposure through scattering, e.g., from engine components, from the martian air, or from the martian soil.

Table 6-6: Conservative Radiation Dose-Rate Limits With 10% EVA Time

In Vehicle 6 meters from Reactor Center	**0.775 mrem/hr**
Outdoors 6 meters from Reactor Center	**88.600 mrem/hr**

CHAPTER 7
7 INITIAL DISCRETE ORDINATES SHIELDING STUDIES

7.1 Summary

Herein, spherical shielding configurations are investigated to approximately estimate the minimum shielding mass needed to reduce radiation leakage to the range of radiation dose exposure limits identified in the previous chapter.

7.2 Computer Codes Used

The initial shielding studies of this chapter relied on the same one dimensional discrete ordinates code used previously for initial criticality studies, i.e., XSDRNPM of the SCALE system of codes. However, in all the runs of the present chapter, the SAS1X sequence of the SCALE module, SAS1 (**S**hielding **A**nalysis **S**equence Number **1**) was used to access XSDRNPM. In the SAS1X sequence, the computation first uses SCALE's Material Information Processor to map from material compositions and problem geometry to the input needed for calls to SCALE's BONAMI and NITAWL modules, which respectively perform self-shielding calculations for materials having Bonderenko data and Nordheim resonance self-shielding corrections for materials having resonance parameters. The XSDRNPM module is then accessed twice to perform criticality and shielding calculations, and finally SCALE's XSDOSE functional module uses the resulting surface angular fluxes to calculate dose rates outside the shield.

In all the runs discussed in this chapter the central reactor sphere was modeled as having a radius of 8.77 cm and a composition matching the final runs of Chapter 5 as listed in Table 7-1. This central reactor core was surrounded by spherical shells with varying thicknesses and compositions in the different shielding cases examined.

The cross sections used in the runs of this chapter were all taken from the 27n/18g coupled ENDF/B-IV library. In addition to modeling neutron production, propagation and absorption, this library includes cross sections modeling the production of gamma ray photons both by fission and by neutron interactions such as inelastic scattering and absorption, and also contains cross sections for modeling the propagation of the resulting gamma rays through shielding material. Thus, by using this library the SAS1X module represents all processes important for the combined shielding of both neutrons and gamma rays.

Table 7-1: Assumed Reactor Core Materials, Densities, and Volume Fractions

Material	Density (g/cm3)	Volume Fraction
UN	14.3	0.8684
Li-7	0.045	0.0778
W	19.4	0.0354
Re	19.4	0.0184

The following sample SAS1X print file excerpt gives radiation doses at R=600 cm.

mt= 9029 ansi standard neutron flux-to-dose-rate factors (rem/hr)/(neut/cm**2/sec)

grp	scalar flux	dose factor	calculated dose
1	4.202E-13	1.492E-04	6.267E-17
2	1.109E-12	1.446E-04	1.604E-16
3	1.052E-12	1.270E-04	1.336E-16
4	9.383E-13	1.281E-04	1.202E-16
5	3.525E-12	1.298E-04	4.575E-16
6	3.842E-11	1.028E-04	3.950E-15
7	5.125E-11	5.118E-05	2.623E-15
8	1.921E-11	1.232E-05	2.367E-16
9	2.866E-12	3.837E-06	1.100E-17
10	9.660E-13	3.725E-06	3.598E-18
11	5.225E-15	4.015E-06	2.098E-20
12	1.556E-17	4.293E-06	6.680E-23
13	8.358E-19	4.474E-06	3.740E-24
14	1.496E-19	4.568E-06	6.835E-25
15	2.472E-20	4.558E-06	1.127E-25
16	8.230E-21	4.519E-06	3.719E-26
17	2.432E-21	4.488E-06	1.092E-26
18	2.454E-21	4.466E-06	1.096E-26
19	5.038E-21	4.435E-06	2.234E-26
20	6.978E-21	4.327E-06	3.019E-26
21	1.623E-21	4.197E-06	6.811E-27
22	1.867E-21	4.098E-06	7.651E-27
23	3.922E-21	3.839E-06	1.506E-26
24	1.700E-21	3.675E-06	6.249E-27
25	1.303E-22	3.675E-06	4.787E-28
26	8.774E-24	3.675E-06	3.224E-29
27	7.783E-26	3.675E-06	2.860E-31
			7.758E-15 (total)

detector 4 (r= 6.000E+02)

mt= 9504 ansi standard gamma-ray flux-to-dose-rate factors (rem/hr)/(photons/cm**2/sec)

grp	scalar flux	dose factor	calculated dose
28	3.423E-17	8.772E-06	3.003E-22
29	2.299E-15	7.478E-06	1.719E-20
30	1.116E-13	6.375E-06	7.111E-19
31	3.813E-13	5.414E-06	2.064E-18
32	1.367E-12	4.622E-06	6.320E-18
33	1.457E-12	3.960E-06	5.768E-18
34	2.306E-12	3.469E-06	7.998E-18
35	2.082E-12	3.019E-06	6.286E-18
36	2.418E-12	2.628E-06	6.353E-18
37	2.582E-12	2.205E-06	5.693E-18
38	1.547E-12	1.833E-06	2.835E-18
39	1.283E-12	1.523E-06	1.953E-18
40	1.315E-12	1.172E-06	1.541E-18
41	2.642E-13	8.759E-07	2.314E-19
42	1.222E-13	6.306E-07	7.705E-20
43	5.337E-14	3.834E-07	2.046E-20
44	2.006E-15	2.669E-07	5.354E-22
45	1.569E-16	9.347E-07	1.466E-22
			4.787E-17 (total)

As stated in the SAS1 portion of the SCALE manual, after converting to dose units the calculated results are "given in rem/hour normalized to a fission source rate of one neutron born per second". The neutron and gamma doserates appearing in the sample SAS1X printout of the previous page must therefore be interpreted by a post-run manual calculation. To convert from its output quantities, (rem/hour)/(fission neutrons/second), to (rem/hour), it is necessary to multiply by (fission neutrons born/second), or equivalently to multiply by

$$\frac{\left(\frac{\text{neutrons born}}{\text{fission}}\right)}{\left(\frac{\text{energy released}}{\text{fission}}\right)}(\text{reactor power})$$

The ratio of neutrons born per fission (frequently called ν) is typically between 2.0 and 3.0 but depends on the energy spectrum of the neutrons causing fission. For SAS1X cases it can be determined from the fine group summary portion of the print file. For the sample printout case of the preceding page, this lists the fission source neutron rates in the inner zone 1 region as being nonzero for each of the first 27 groups (i.e., the neutron groups) but zero for the next 18 groups (i.e., the gamma groups) while summing to a total of exactly 1. The fission rate sum is listed as 4.000E-01. Therefore for this case the value of neutrons born per fission is 1/0.4=2.500.

Since about 1 joule of energy is released by $3.1*10^{10}$ fissions [Stacey 2001, 13] the factor for a 1 MWt reactor with the Table 7-1 case energy spectrum is as follows:

$$\frac{\left(\frac{\text{neutrons born}}{\text{fission}}\right)}{\left(\frac{\text{energy released}}{\text{fission}}\right)}(\text{reactor power}) = \frac{\left(2.5\frac{\text{neutrons born}}{\text{fission}}\right)}{\left(\frac{1\,\text{Joule}}{3.1*10^{10}\,\text{fissions}}\right)}(10^6\ \text{watts}) =$$

$$= 7.75*10^{16}\,\frac{\text{neutrons}}{\text{second}}$$

Applying this factor to the sample printout's doserate totals, results are as follows:

[7.758E-15 [(rem/hour)/(fission neutrons/second)]*(7.75E16 neutrons/second)=
= 601.25 rem/hour dose from neutrons
= 6012.5 mSv h^{-1} from neutrons
and
[4.787E-17 [(rem/hour)/(fissionneutrons/second)]*(7.75E16 neutrons/second)=
= 3.710 rem/hour dose from gamma rays
= 37.10 mSv h^{-1} from gamma rays

Total Doserate = 6012.5 mSv h^{-1} + 37.10 mSv h^{-1}
= 6.0496 Sv h^{-1}

These calculated dose rates from neutron and gamma ray components of total dose are very much higher than the 0.01375 mSv h^{-1} acceptable average total dose rate discussed in the previous chapter. Thus, this case represents insufficient shielding for a 1 MWt reactor power level.

To match the Table 6-5 acceptable average total radiation dose rate for a 1 MWt reactor power level of the same energy spectrum as in this samplecase, the total of the print file's values for normalized neutron plus gamma doses should be:

(1.375E-3 rem/h)/(7.75E16 neutrons/second)=
=1.774E-20 [(rem/hour)/(fission neutrons/second)]

This should be reduced by a factor of 10 to 1.774E-21 for a 10 MWt reactor but may be increased by a factor of 10 to 1.774E-19 for a 0.1 MWt reactor.

For the lower dose rate of 0.00775 mSv h^{-1} suggested in the previous chapter's Table 6-6 for the pressurized vehicle, if the reactor power were 1 MWt then the print file's values for normalized neutron plus gamma doses should be:
(0.775E-3 rem/h)/(7.75E16 neutrons/second)=
=1.000E-20 [(rem/hour)/(fissionneutrons/second)]

Similarly, the dose rate of 0.886 mSv h^{-1} suggested for outdoor locations 6 meters from the 1 MWt reactor's center would correspond to print file neutron plus gamma dose totals of
(0.0886 rem/h)/(7.75E16 neutrons/second)=
=**1.143E-18 [(rem/hour)/(fission neutrons/second)]**

For many of the sequences of SAS1X runs described in the present chapter, the value of 1.000E-20 was adopted as the target value for total neutron+gamma normalized radiation dose at a location R=6 meters from the reactor center. However, this normalized total dose rate value was only used offline to help decide what changes to make in subsequent SAS1X runs. It was never entered in any way as part of the SAS1X input data.

7.3 The Bare Reactor Case (No Shielding)

With the R=8.77 cm spherical reactor not surrounded by any reflector material, the k-effective value calculated by SAS1X drops to 0.72296. However, regardless of this low criticality value, SAS1X uses its resulting calculated neutron energy spectrum to determine the external neutron and gamma dose rates per fission neutron per second. Results for R=6 m from the reactor center are listed in Table 7-2. It is noteworthy that for this "bare reactor case", the total unshielded doserate 6 meters from the center of the reactor producing 1 MW of thermal poiwer would be 12.2 million Sievert per hour.

Table 7-2: Radiation Dose 6 meters From Center Of Unshielded Bare Spherical Reactor

	Normalized Dose Rate (rem/hour)/(fission neutrons/second)	**Dose mSv h^{-1} (if 1MWt reactor power)**
From Neutrons	**1.563E-11**	**1.21E+7**
From Gamma Rays	**7.719E-14**	**5.98 E+4**
Total	**N/A**	**1.22 E7**

Thus, without shielding a 1 MWt reactor would cause a 6 Sv lethal dose 6 meters away in less than 2 seconds. To reduce this intensity to a safe 0.01375 mSv h^{-1} level requires shielding to attenuate the doserate by almost a billion. To alternatively reduce the normalized dose rate sum to 1.000E-20 requires an attenuation factor more than a billion. These are large attenuation factors and may be challenging to obtain in a compact shield design. Shield designs with such large attenuation factors are sometimes called deep penetration designs. They require special care since they could be compromised by small errors either in calculation or in physical implementation. For instance, an uncertainty of only about 10% in shield material cross sections would imply a factor of 10 uncertainty in the shield's radiation leakage. Also, any cracks in the shield construction may defeat the shield's purpose.

7.4 Performance of Single Material Shields

Although texts and other publications report that single material shields perform poorly in comparison with layered shields made from combinations of multiple materials, they were examined first here because of their simplicity. For each material a series of shields was analyzed in which the R=8.77 cm reactor core was surrounded by progressively thicker spherical shells of the shield material. The tables and plots below summarize resulting normalized radiation dose rates at R=6 m, the shield masses, and the k-effective values.

7.4.1 Beryllium

Table 7-3 summarizes results of SAS1X runs in which shielding is provided by a spherical shell of beryllium surrounding a spherical reactor of R=8.77 cm radius. This doserate vs. shield size data are plotted in Figure 7-1. As per Table 7-3 and Figure 7-1, the neutron dose rapidly falls by almost two orders of magnitude as the outer radius of the beryllium is increased to about 50 cm, then it falls much more slowly in tandem with the gamma ray dose. The normalized total dose at R=6 m reaches the 1.143E-18 (rem/hour)/(fission neutrons/second) level suggested for outdoor locations when the Be shielding extends all the way to R=4.95 m. The mass of such a large shield would be 940 tonnes, which is absurdly too large to be practical for a surface vehicle on Mars. The lower normalized dose at R=6 m of 1.000E-20 (rem/hour)/(fission neutrons/second) suggested for locations inside the pressurized surface vehicle's cabin cannot be achieved at all using only beryllium, even if the spherical shell extended all the way to R=6 m which would require a shield mass of 1,674 tonnes.

Table 7-3: Radiation Leakage Vs Size For a Radiation Shield of Pure Beryllium

Outer Radius of Shell (cm)	neutron dose	gamma dose	total dose	k-effective	shield mass (kg)
8.77	1.563E-11	7.719E-14	1.571E-11	0.72296	0
10	1.400E-11	9.098E-14	1.409E-11	0.80347	2.5
15	8.247E-12	1.144E-13	8.361E-12	0.99812	20.9
20	4.501E-12	1.217E-13	4.623E-12	1.09899	56.8
30	1.292E-12	1.195E-13	1.412E-12	1.20435	204.0
40	4.786E-13	1.141E-13	5.927E-13	1.24805	490.7
50	2.768E-13	1.092E-13	3.860E-13	1.26672	963.4
60	2.030E-13	1.026E-13	3.056E-13	1.27527	1668.6
70	1.594E-13	9.394E-14	2.533E-13	1.27931	2652.8
80	1.260E-13	8.370E-14	2.097E-13	1.28123	3962.4
90	9.885E-14	7.269E-14	1.715E-13	1.28215	5644.0
100	7.678E-14	6.174E-14	1.385E-13	1.28259	7744.0
120	4.502E-14	4.218E-14	8.720E-14	1.28291	13385.5
150	1.915E-14	2.156E-14	4.071E-14	1.28299	26148.5
180	7.836E-15	1.020E-14	1.804E-14	1.28299	45188.5
200	4.304E-15	6.097E-15	1.040E-14	1.28299	61988.9
300	1.813E-16	3.506E-16	5.319E-16	1.28299	209224.8
400	6.949E-18	1.662E-17	2.357E-17	1.28298	495947.5
500	2.605E-19	7.165E-19	9.770E-19	1.28298	968652.5
600	1.175E-20	3.031E-20	4.206E-20	1.28298	1673835.3

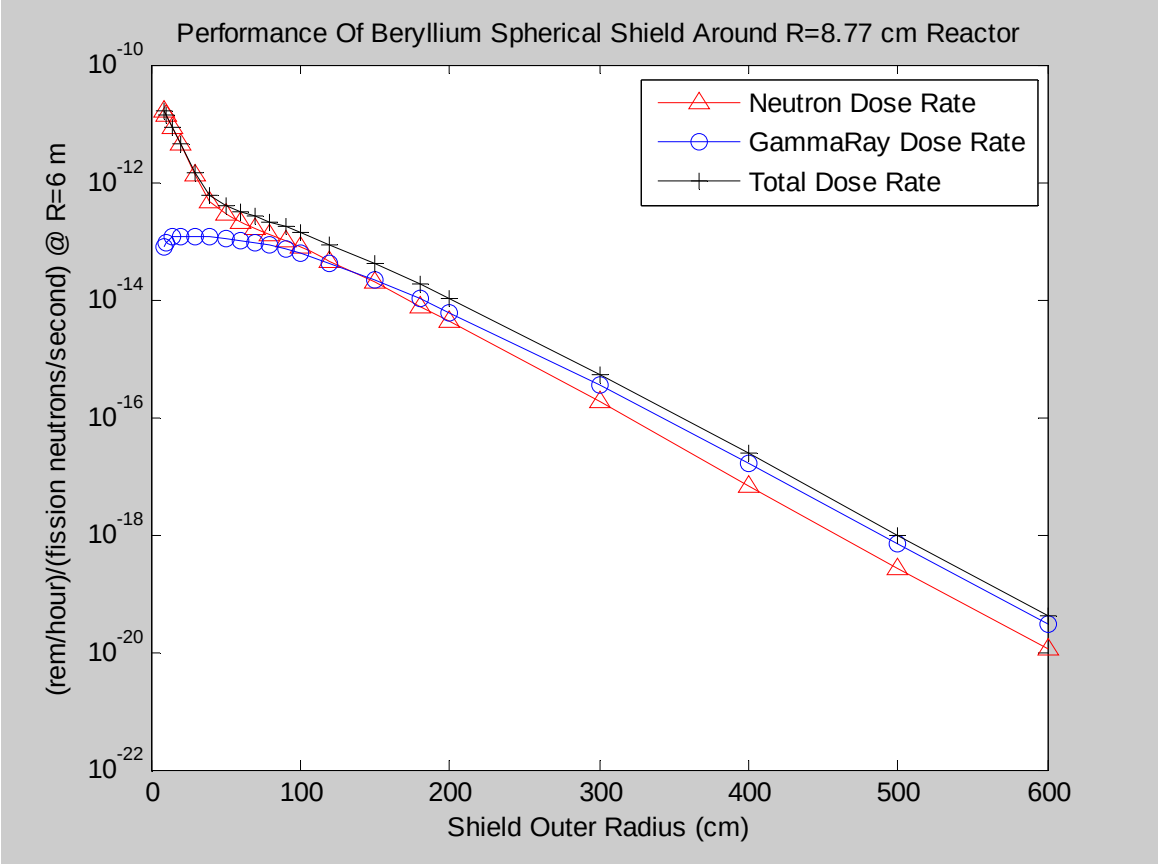


Figure 7-1: Performance of Beryllium Single Material Shield

7.4.2 Boron-10 Carbide

Table 7-4 summarizes results of SAS1X runs in which shielding is provided by a spherical shell of boron carbide (B_4C) in which the boron is entirely the boron-10 isotope. The radiation dose dependence on shield size is also plotted in Figure 7-2. The normalized total dose at R=6 m reaches the 1.143E-18 (rem/hour)/(fission neutrons/second) level previously suggested for outdoor locations with a 1 MWt reactor when the $^{10}B_4C$ shielding extends to R=1.71 m, for which the shield mass would be 52.8 tonnes. The dose at R=6 m reaches the 1.000E-20 (rem/hour)/(fission neutrons/second) level suggested as the effective average inside the pressurized vehicle when the $^{10}B_4C$ shielding extends to R=2.52 m. The mass of a $^{10}B_4C$ shield extending to 2.52 m would be 168.9 tonnes.

From Figure 7-2 the neutron dose rate shows an exponential dependence on shield thickness while the gamma ray dose dependence shows a small slope change. The slope change may be due to gamma ray production by neutron absorption.

7.4.3 Boron-10

Table 7-5 summarizes results of SAS1X runs in which shielding is provided by a spherical shell of elemental boron composed purely of the boron-10 isotope. The radiation dose dependence on shield size is also plotted in Figure 7-3. The results are very similar to the boron carbide case. Pure boron-10 has slightly better neutron attenuation due to its higher boron-10 atom density but both are excellent. Boron carbide has slightly better gamma ray attenuation than boron, but both are poor.

7.4.4 Carbon (Graphite)

The SAS1X results for a single material shield made of graphite carbon are shown in Table 7-6 and Figure 7-4. With graphite, after an initial rapid decrease in neutron intensity for thin shields, the neutron and gamma intensities then diminish slowly in tandem with each other as the shield thickness is increased further. Although graphite has been found useful as a neutron moderator and/or reflector material in some nuclear reactor designs, by itself it apparently is not an effective shield for either neutron or gamma radiations. Even if a spherical shell of graphite extended all the way to a radius of R=6 meters (requiring 1,990 tonnes of carbon), its radiation leakage would still be hundreds of times higher than the 1.143E-18 (rem/hour)/(fission neutrons/second) level previously suggested for outdoor locations at that distance from the reactor's center. Graphite is only included here because it has been applied as a high temperature structural material at temperatures up to 2773 Kelvins (2500C), which exceeds maximum temperatures contemplated for this nuclear engine application Graphite is strong, it has anisotropic thermal conduction properties which may be exploited for thermal insulation purposes, and it has a comparatively low mass density.

Table 7-4: Radiation Leakage Vs Size For Radiation Shield of BoronCarbide Using B-10

Outer Radius of Shell (cm)	neutron dose	gamma dose	total dose	k-effective	shield mass (kg)
8.77	1.563E-11	7.719E-14	1.571E-11	0.72296	0
10	1.264E-11	9.294E-14	1.273E-11	0.78179	3.4
15	5.095E-12	9.384E-14	5.189E-12	0.82874	28.5
20	2.067E-12	7.022E-14	2.137E-12	0.83164	77.3
30	3.443E-13	3.053E-14	3.748E-13	0.83192	277.9
40	5.838E-14	1.200E-14	7.038E-14	0.83192	668.4
50	1.016E-14	4.772E-15	1.493E-14	0.83192	1312.3
60	1.824E-15	2.007E-15	3.831E-15	0.83192	2272.9
70	3.371E-16	8.977E-16	1.235E-15	0.83192	3613.5
80	6.396E-17	4.221E-16	4.861E-16	0.83192	5397.4
90	1.241E-17	2.059E-16	2.183E-16	0.83193	7688.0
100	2.457E-18	1.032E-16	1.057E-16	0.83193	10548.6
120	1.008E-19	2.738E-17	2.748E-17	0.83193	18233.2
150	9.001E-22	4.103E-18	4.104E-18	0.83192	35618.5
180	8.363E-24	6.607E-19	6.607E-19	0.83192	61554.0
200	3.758E-25	2.059E-19	2.059E-19	0.83192	84438.9
300	6.218E-32	6.531E-22	6.531E-22	0.83192	284998.2
400	9.497E-39	2.469E-24	2.469E-24	0.83191	675561.0

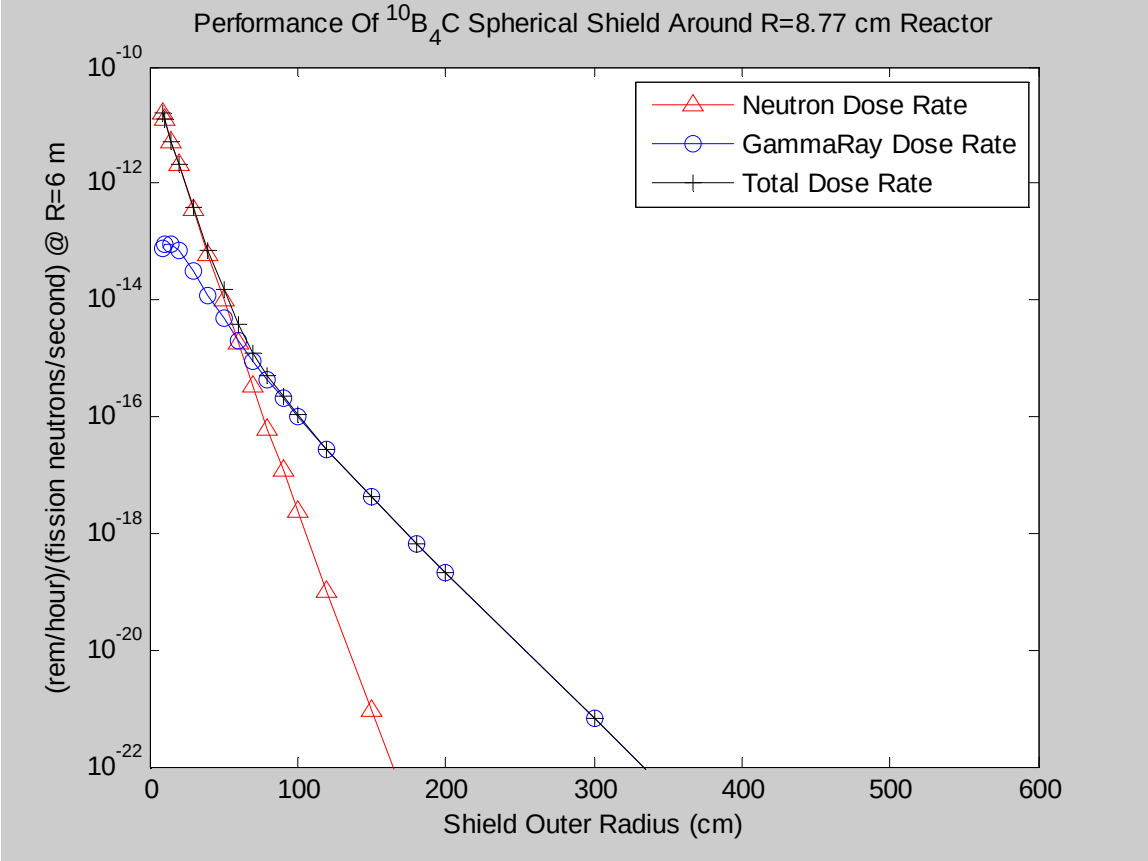


Figure 7-2: Performance of $^{10}B_4C$ Single Material Shield

Table 7-5: Radiation Leakage Vs Size For a Radiation Shield of Elemental Boron Using B-10

Outer Radius of Shell (cm)	neutron dose	gamma dose	total dose	k-effective	shield mass (kg)
8.77	1.563E-11	7.719E-14	1.571E-11	0.72296	0
10	1.242E-11	9.389E-14	1.251E-11	0.77619	3.2
15	4.784E-12	9.594E-14	4.880E-12	0.81471	26.8
20	1.900E-12	7.298E-14	1.973E-12	0.81684	72.7
30	3.121E-13	3.335E-14	3.455E-13	0.81703	261.3
40	5.331E-14	1.378E-14	6.709E-14	0.81704	628.7
50	9.429E-15	5.705E-15	1.513E-14	0.81704	1234.2
60	1.717E-15	2.471E-15	4.188E-15	0.81704	2137.6
70	3.196E-16	1.131E-15	1.451E-15	0.81704	3398.4
80	6.041E-17	5.443E-16	6.047E-16	0.81704	5076.1
90	1.154E-17	2.721E-16	2.836E-16	0.81704	7230.4
100	2.227E-18	1.400E-16	1.422E-16	0.81704	9920.7
120	8.471E-20	3.939E-17	3.947E-17	0.81704	17147.9
150	6.577E-22	6.515E-18	6.516E-18	0.81704	33498.4
180	5.313E-24	1.167E-18	1.167E-18	0.81704	57890.1
200	2.190E-25	3.916E-19	3.916E-19	0.81703	79412.8
300	2.504E-32	1.792E-21	1.792E-21	0.81703	268034.0
400	2.733E-39	9.660E-24	9.660E-24	0.81703	635349.0

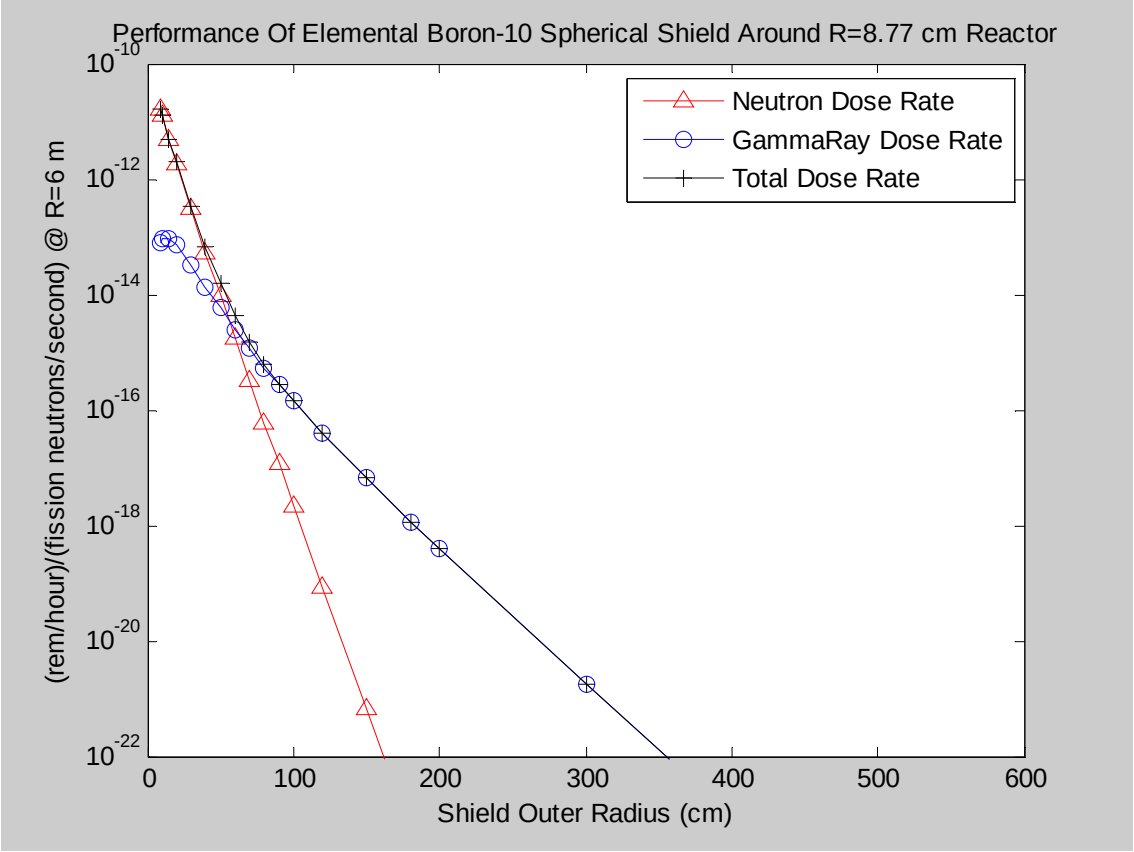


Figure 7-3: Performance of Boron-10 Single Material Shield

Table 7-6: Radiation Leakage Vs Size For a Radiation Shield of Graphite Carbon

Outer Radius of Shell (cm)	neutron dose	gamma dose	total dose	k-effective	shield mass (kg)
8.77	1.563E-11	7.719E-14	1.57072E-11	0.72296	0
10	1.450E-11	8.491E-14	1.45849E-11	0.78596	3.0
20	7.276E-12	7.628E-14	7.35228E-12	1.00528	67.5
30	1.268E-12	3.011E-14	1.29811E-12	1.13048	242.6
50	6.336E-13	2.034E-14	6.5394E-13	1.15614	1145.7
100	2.387E-13	1.008E-14	2.4878E-13	1.19518	9209.1
150	1.480E-13	6.139E-15	1.54139E-13	1.20085	31095.6
200	8.529E-14	3.474E-15	8.8764E-14	1.20182	73716.5
300	2.386E-14	9.521E-16	2.48121E-14	1.20202	248807.9
400	6.084E-15	2.391E-16	6.3231E-15	1.20202	589775.4
500	1.518E-15	5.858E-17	1.57658E-15	1.20202	1151911.1
600	4.543E-16	1.608E-17	4.7038E-16	1.20202	1990506.9

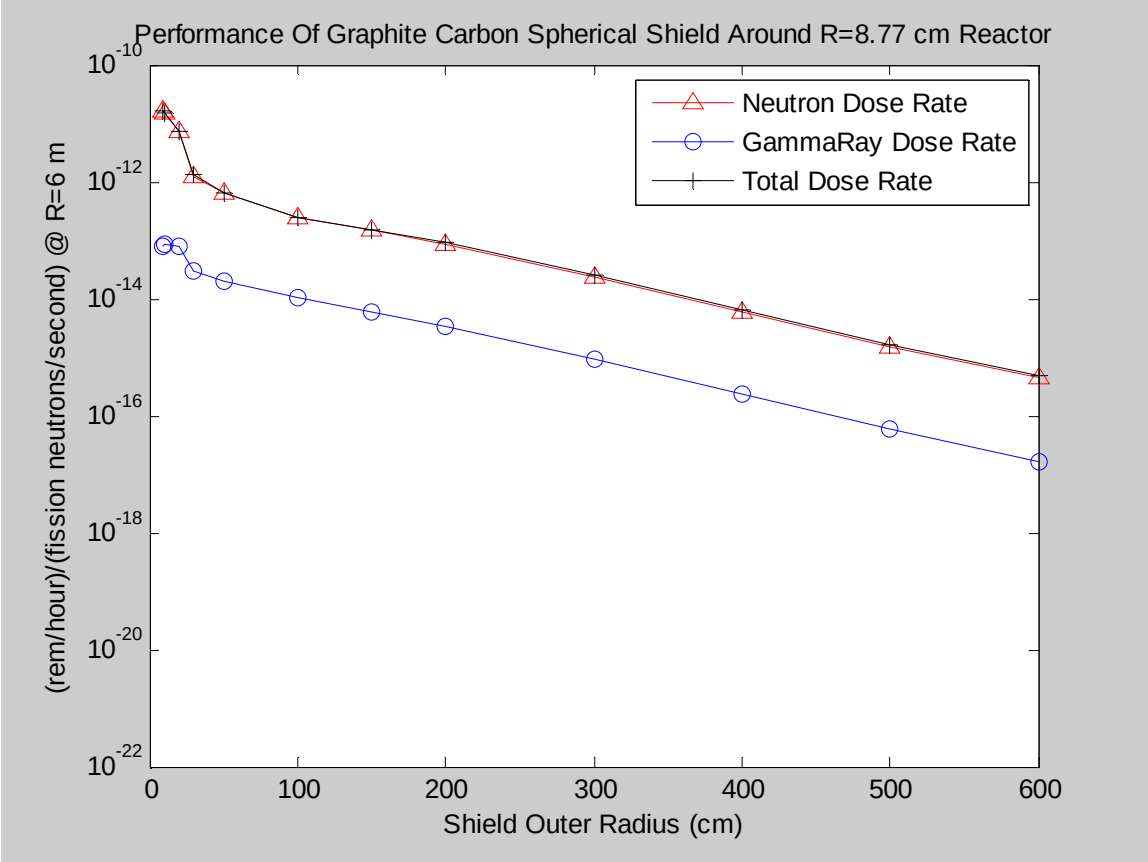


Figure 7-4: Performance of Graphite Carbon Single Material Shield

7.4.5 Iron

The SAS1X results for a single material shield made of iron are shown in Table 7-7 and Figure 7-5. As shown, iron is not particularly effective as a single material shield since thousands of tons would be needed.

7.4.6 Liquid Lead (Pb)

The SAS1X results for a single material shield of liquid lead are shown in Table 7-8 and Figure 7-6. Its superior attenuation of gamma rays is masked here by gamma ray production caused by inelastic scattering and absorption interactions with neutrons.

7.4.7 Lithium Hydride Solid (Natural Lithium Isotope Mix)

The SAS1X results for a single material shield made of solid lithium hydride using the natural isotopic mix of lithium are shown in Table 7-9 and Figure 7-7. This material is almost transparent for gamma rays but is an excellent attenuator of neutrons.

7.4.8 Lithium Hydride Solid (Lithium-6 Isotope Only)

The SAS1X results for a shield made of lithium hydride solid using only the lithium-6 isotope are shown in Table 7-10 and Figure 7-8. It is similar to the natural lithium hydride case, a slightly better neutron attenuator for thin shields and slightly poorer for thick shields. Their difference may be due to better inelastic scattering by Li-7.

7.4.9 Polyethylene

The SAS1X results for a single material shield made of polyethylene plastic are shown in Table 7-11 and Figure 7-9.

7.4.10 Tungsten

The SAS1X results for a single material shield made of tungsten are shown in Table 7-12 and Figure 7-10. Per unit thickness, tungsten is the most effective inelastic scatterer and it is also an effective absorber of both neutrons and gammas. However, it is also very dense so even a thin tungsten shield may be prohibitively massive.

7.4.11 Uranium-238

The SAS1X results for uranium-238 are shown in Table 7-13 and Figure 7-11. Although this is the best material to use as a gamma ray shield when neutrons are absent, it produces gamma rays internally by inelastic scattering absorbing neutrons. For uranium-238 shields thicker than 2 meters the neutron flux first stops decreasing as the shield is made thicker, then increases between R=3 m and R=4 m. This may be due to fast fission.

Table 7-7: Radiation Leakage Vs Size For a Radiation Shield of Pure Iron

Outer Radius of Shell (cm)	neutron dose	gamma dose	total dose	k-effective	shield mass (kg)
8.77	1.563E-11	7.719E-14	1.571E-11	0.72296	0
10	1.474E-11	7.663E-14	1.482E-11	0.77249	10.7
15	1.208E-11	4.376E-14	1.212E-11	0.87871	88.9
20	1.006E-11	2.259E-14	1.008E-11	0.93143	241.2
30	7.049E-12	8.875E-15	7.058E-12	0.98386	866.7
40	4.998E-12	6.281E-15	5.004E-12	1.00689	2084.9
50	3.559E-12	5.628E-15	3.565E-12	1.01735	4093.3
60	2.536E-12	5.132E-15	2.541E-12	1.02200	7089.4
70	1.802E-12	4.523E-15	1.807E-12	1.02398	11270.7
80	1.275E-12	3.833E-15	1.279E-12	1.02476	16834.8
90	8.978E-13	3.141E-15	9.009E-13	1.02502	23979.3
100	6.292E-13	2.504E-15	6.317E-13	1.02505	32901.7
120	3.049E-13	1.490E-15	3.064E-13	1.02493	56870.3
150	1.002E-13	6.090E-16	1.008E-13	1.02472	111095.9
180	3.244E-14	2.287E-16	3.267E-14	1.02457	191989.9
200	1.539E-14	1.167E-16	1.551E-14	1.02450	263368.9
300	3.380E-16	3.097E-18	3.411E-16	1.02431	888922.8
400	7.246E-18	7.019E-20	7.316E-18	1.02422	2107106.8
500	1.560E-19	1.528E-21	1.575E-19	1.02418	4115464.2
600	4.079E-21	3.861E-23	4.118E-21	1.02415	7111538.3

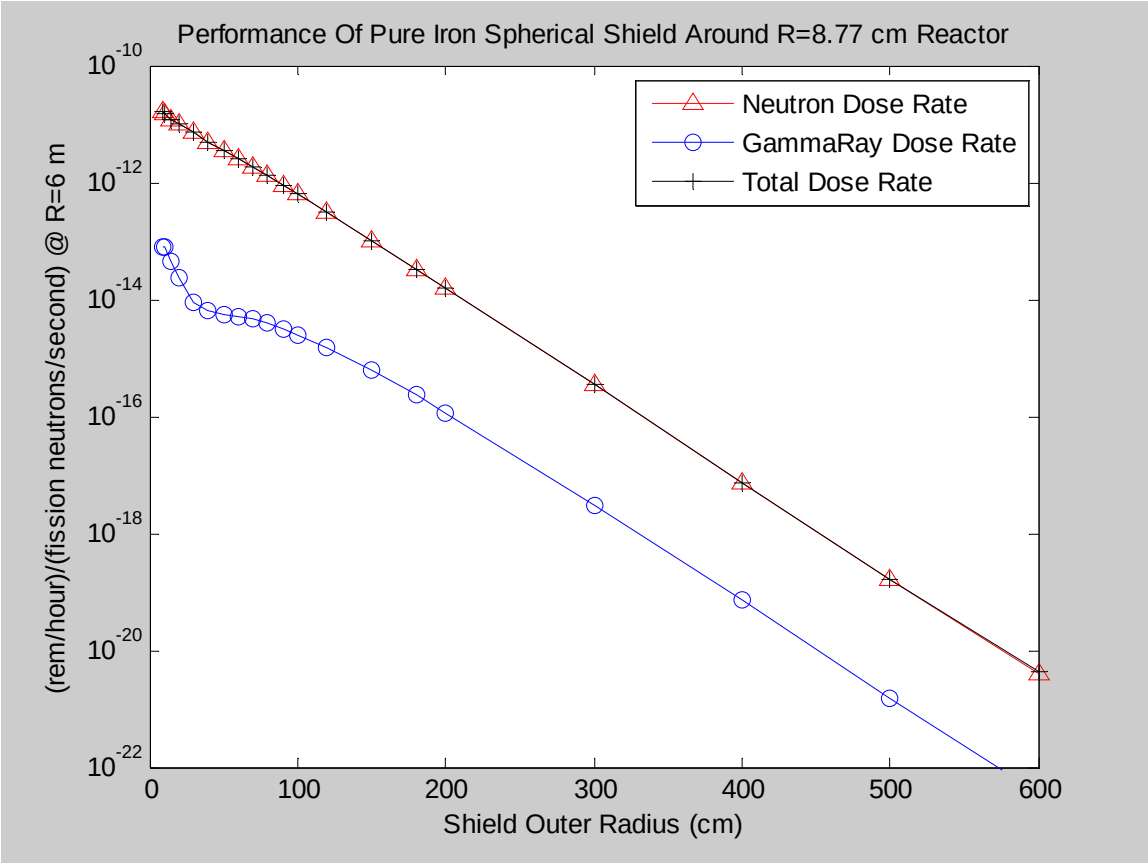


Figure 7-5: Performance of Pure Iron Single Material Shield

Table 7-8: Radiation Leakage Vs Size For a Radiation Shield of Liquid Lead (Pb)

Outer Radius of Shell (cm)	neutron dose	gamma dose	total dose	k-effective	shield mass (kg)
8.77	1.563E-11	7.719E-14	1.571E-11	0.72296	0
10	1.515E-11	4.563E-14	1.520E-11	0.75685	13.7
15	1.392E-11	1.064E-14	1.393E-11	0.82847	113.7
20	1.306E-11	4.799E-15	1.306E-11	0.8645	308.4
30	1.163E-11	2.129E-15	1.163E-11	0.90344	1108.2
40	1.036E-11	1.150E-15	1.036E-11	0.92522	2665.8
50	9.118E-12	6.921E-16	9.119E-12	0.93963	5233.8
60	8.002E-12	5.011E-16	8.003E-12	0.94900	9064.6
70	6.954E-12	4.098E-16	6.954E-12	0.95598	14411.0
80	5.977E-12	3.676E-16	5.977E-12	0.96128	21525.4
100	4.300E-12	3.468E-16	4.300E-12	0.96831	42068.9
120	2.998E-12	3.521E-16	2.998E-12	0.97216	72715.8
150	1.674E-12	3.493E-16	1.674E-12	0.97458	142050.1
180	9.104E-13	3.152E-16	9.107E-13	0.97536	245483.3
200	6.08E-13	2.82E-16	6.081E-13	0.97551	336750.3
300	7.05E-14	8.79E-17	7.058E-14	0.99558	1136599.8
400	7.14E-15	1.47E-17	7.152E-15	0.97557	2694201.5
500	6.30E-16	1.71E-18	6.318E-16	0.97556	5262139.3
600	5.92E-17	1.91E-19	5.940E-17	0.97556	9092997.4

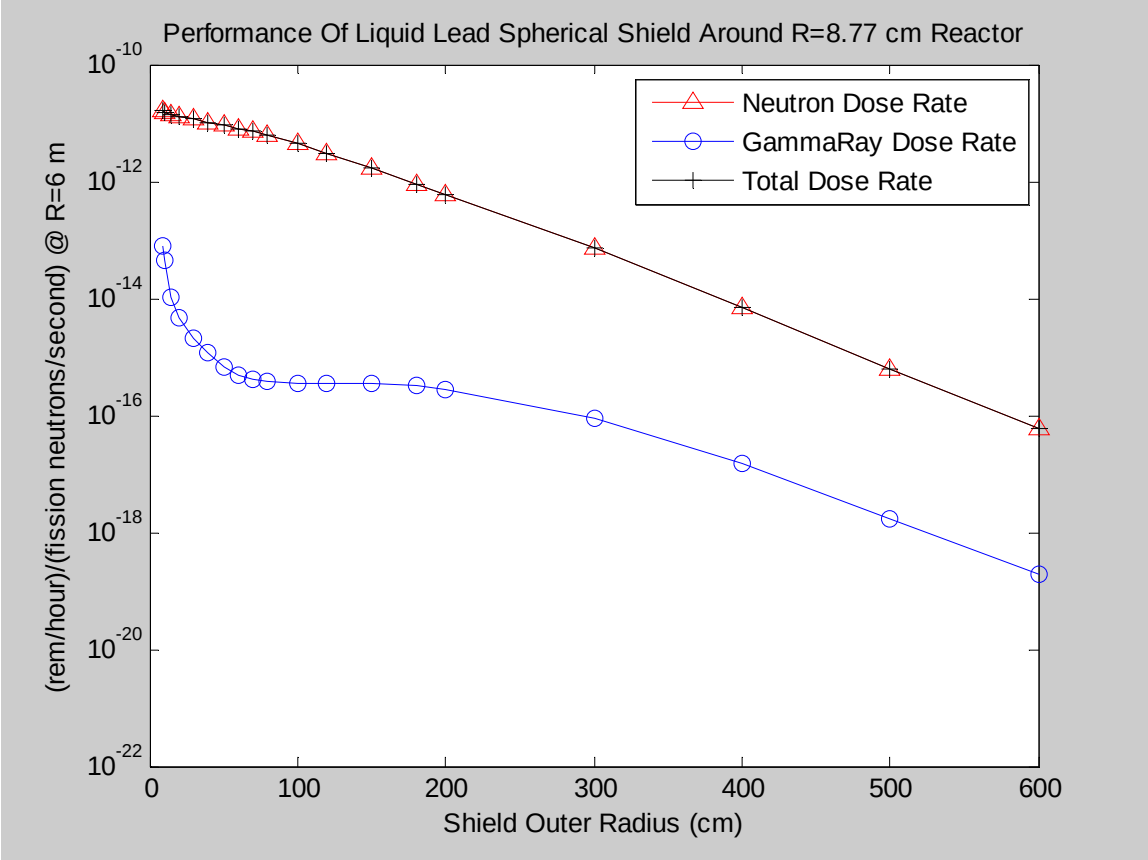


Figure 7-6: Performance of Liquid Lead Single Material Shield

Table 7-9: Radiation Leakage Vs Size For a Radiation Shield of Solid Lithium Hydride (Natural Isotopic Mix)

Outer Radius of Shell (cm)	neutron dose	gamma dose	total dose	k-effective	shield mass (kg)
8.77	1.563E-11	7.719E-14	1.571E-11	0.72296	0
10	1.320E-11	9.042E-14	1.329E-11	0.75837	1.0
15	6.183E-12	1.056E-13	6.289E-12	0.80625	7.9
20	2.753E-12	9.964E-14	2.853E-12	0.80964	21.5
30	5.369E-13	8.509E-14	6.220E-13	0.80993	77.2
40	1.096E-13	7.232E-14	1.819E-13	0.80994	185.7
50	2.434E-14	6.152E-14	8.586E-14	0.80994	364.5
60	5.982E-15	5.217E-14	5.815E-14	0.80993	631.4
70	1.619E-15	4.395E-14	4.557E-14	0.80993	1003.8
80	4.709E-16	3.670E-14	3.717E-14	0.80993	1499.3
100	4.469E-17	2.491E-14	2.495E-14	0.80993	2930.2
120	4.454E-18	1.636E-14	1.636E-14	0.80993	5064.8
150	1.392E-19	8.375E-15	8.375E-15	0.80993	9894.0
180	4.238E-21	4.216E-15	4.216E-15	0.80993	17098.3
200	4.13E-22	2.74E-15	2.738E-15	0.80993	23455.2
300	3.07E-27	3.28E-16	3.276E-16	0.80993	79166.2
400	2.08E-32	5.09E-17	5.094E-17	0.80993	187655.8
500	1.37E-37	9.36E-18	9.355E-18	0.80993	366517.2
600	1.02E-42	1.94E-18	1.943E-18	0.80993	633343.1

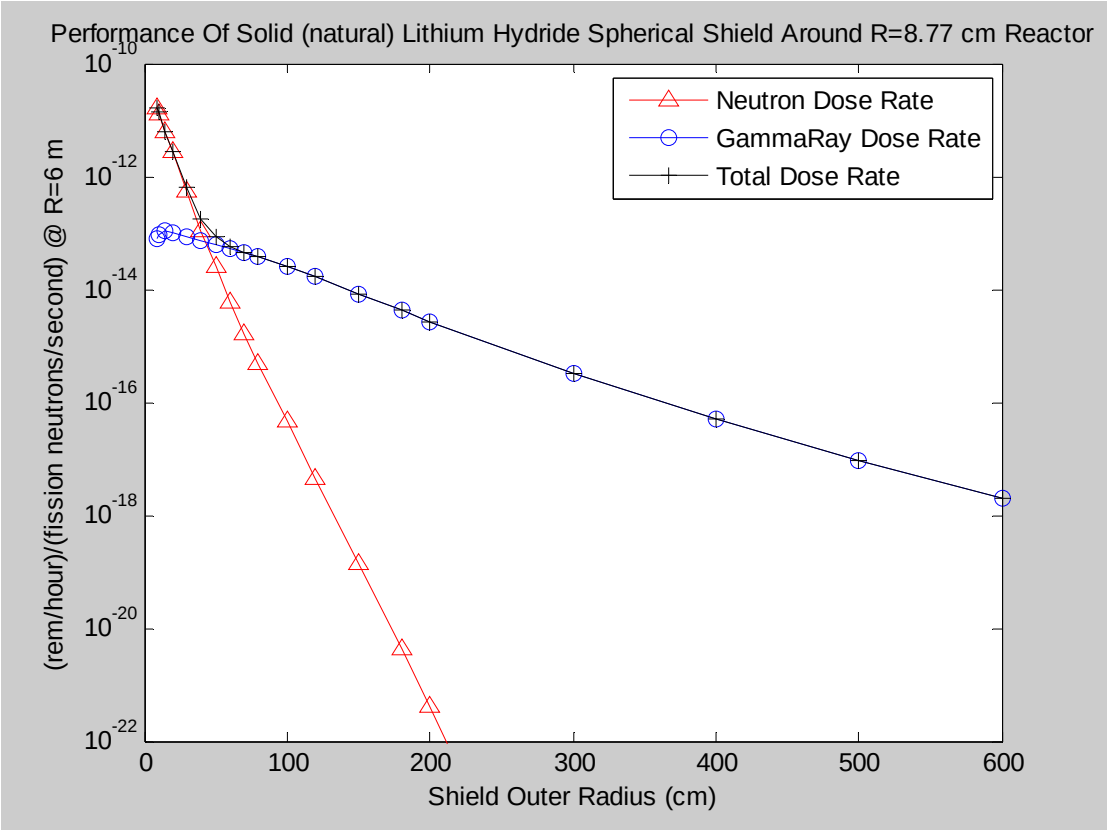


Figure 7-7: Performance of Solid Lithium Hydride Single Material Shield With Natural Isotopic Mix Lithium

Table 7-10: Radiation Leakage Vs Size For a Radiation Shield of Solid Lithium Hydride (Li-6 Isotope Only)

Outer Radius of Shell (cm)	neutron dose	gamma dose	total dose	k-effective	shield mass (kg)
8.77	1.563E-11	7.719E-14	1.571E-11	0.72296	0
10	1.304E-11	8.434E-14	1.312E-11	0.74764	0.8
15	6.099E-12	8.308E-14	6.182E-12	0.76732	6.9
20	2.836E-12	7.648E-14	2.912E-12	0.76836	18.8
30	6.155E-13	6.473E-14	6.802E-13	0.76846	67.6
40	1.369E-13	5.504E-14	1.919E-13	0.76846	162.6
50	3.194E-14	4.689E-14	7.883E-14	0.76846	319.2
60	7.962E-15	3.986E-14	4.782E-14	0.76846	552.9
70	2.139E-15	3.369E-14	3.583E-14	0.76846	879.0
80	6.149E-16	2.826E-14	2.887E-14	0.76846	1312.9
100	5.815E-17	1.940E-14	1.946E-14	0.76846	2566.0
120	5.956E-18	1.293E-14	1.294E-14	0.76846	4435.3
150	1.994E-19	6.769E-15	6.769E-15	0.76846	8664.4
180	6.568E-21	3.481E-15	3.481E-15	0.76846	14973.3
200	6.75E-22	2.29E-15	2.290E-15	0.76846	20540.1
300	6.57E-27	2.86E-16	2.855E-16	0.76846	69326.9
400	5.83E-32	4.56E-17	4.559E-17	0.76846	164332.9
500	5.00E-37	8.59E-18	8.592E-18	0.76846	320964.3
600	4.76E-42	1.83E-18	1.832E-18	0.76847	554627.6

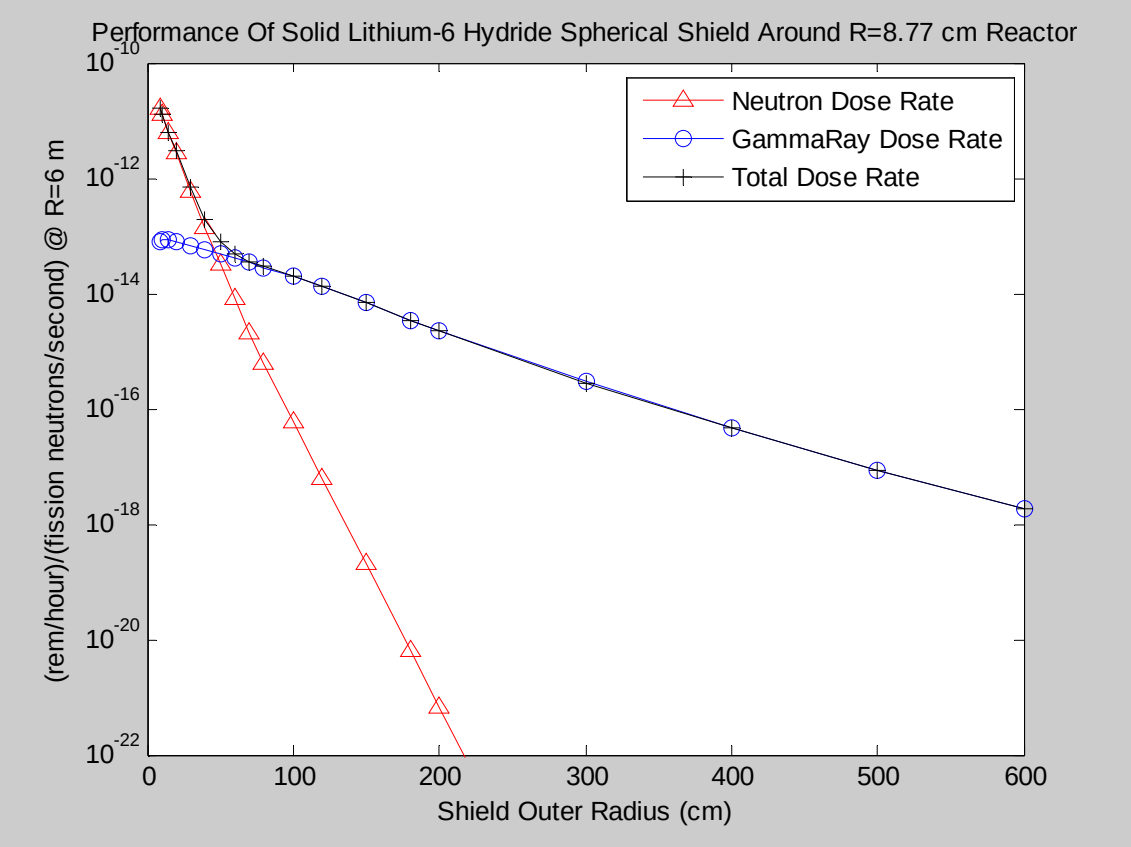


Figure 7-8: Performance of Solid Lithium Hydride Single Material Shield With Lithium-6 Isotope Only

Table 7-11: Radiation Leakage Vs Size For a Radiation Shield of Polyethylene (Plastic)

Outer Radius of Shell (cm)	neutron dose	gamma dose	total dose	k-effective	shield mass (kg)
8.77	1.563E-11	7.719E-14	1.571E-11	0.72296	0
10	1.244E-11	1.027E-13	1.254E-11	0.78219	1.3
15	4.814E-12	2.611E-13	5.075E-12	0.93674	10.4
20	1.863E-12	3.513E-13	2.214E-12	0.95354	28.2
30	3.191E-13	3.303E-13	6.494E-13	0.95445	101.5
40	7.000E-14	2.539E-13	3.239E-13	0.95443	244.0
50	1.862E-14	1.878E-13	2.064E-13	0.95442	479.1
60	5.751E-15	1.367E-13	1.425E-13	0.95441	829.8
70	1.944E-15	9.856E-14	1.005E-13	0.95441	1319.2
80	6.871E-16	7.053E-14	7.122E-14	0.95441	1970.5
100	8.945E-17	3.563E-14	3.572E-14	0.95441	3851.1
120	1.163E-17	1.782E-14	1.783E-14	0.95440	6656.6
150	5.275E-19	6.272E-15	6.273E-15	0.95440	13003.6
180	2.316E-20	2.227E-15	2.227E-15	0.9544	22472.1
200	2.88E-21	1.16E-15	1.156E-15	0.9544	30826.9
300	7.23E-26	4.80E-17	4.797E-17	0.9544	104046.9
400	1.66E-30	2.98E-18	2.983E-18	0.9544	246633.4
500	3.69E-35	2.39E-19	2.388E-19	0.9544	481708.3
600	9.03E-40	2.18E-20	2.179E-20	0.9544	832393.8

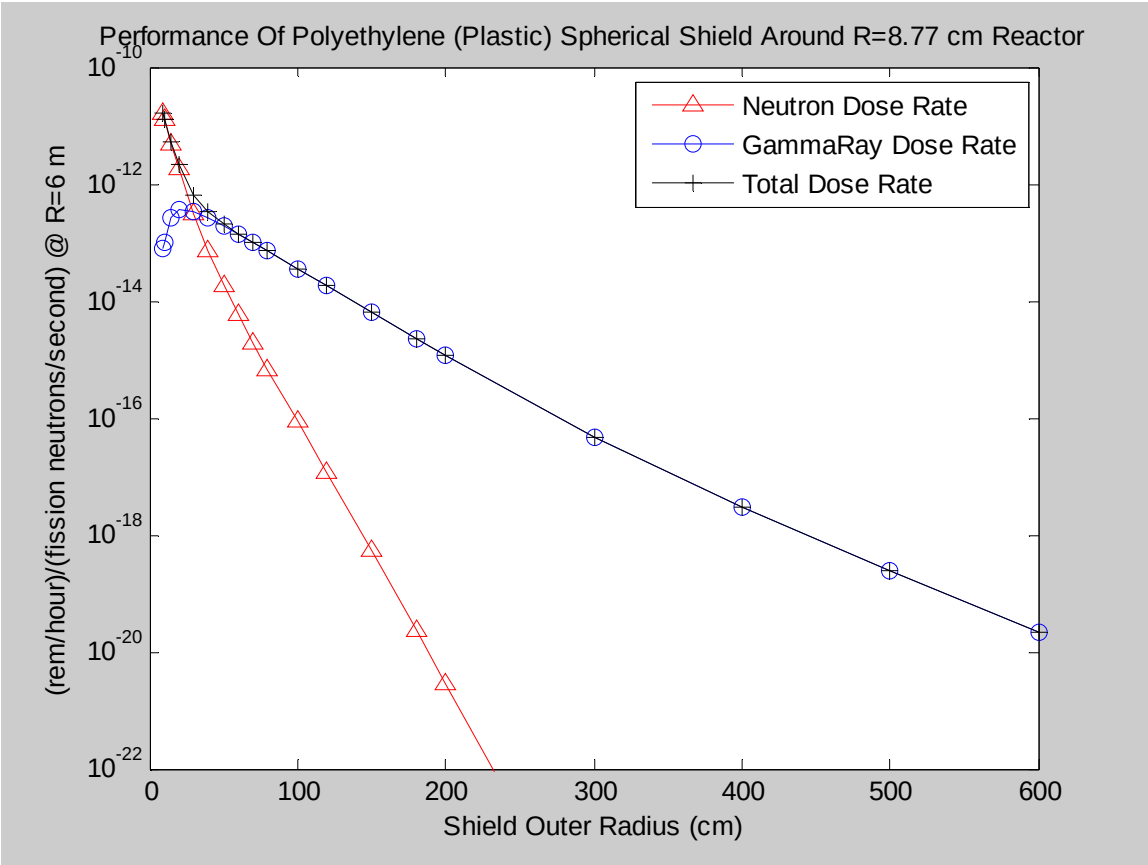


Figure 7-9: Performance of Polyethylene (Plastic) Single Material Shield

Table 7-12: Radiation Leakage Vs Size For a Radiation Shield of Tungsten

Outer Radius of Shell (cm)	neutron dose	gamma dose	total dose	k-effective	shield mass (kg)
8.77	1.563E-11	7.719E-14	1.571E-11	0.72296	0
10	1.401E-11	4.422E-14	1.405E-11	0.79248	26.4
15	9.164E-12	1.386E-14	9.178E-12	0.93601	219.4
20	5.968E-12	8.815E-15	5.977E-12	0.98716	595.3
30	2.389E-12	4.543E-15	2.394E-12	1.01090	2139.3
40	8.946E-13	2.109E-15	8.967E-13	1.01329	5146.0
50	3.188E-13	8.631E-16	3.197E-13	1.01351	10103.0
60	1.098E-13	3.238E-16	1.101E-13	1.01353	17497.9
70	3.685E-14	1.146E-16	3.696E-14	1.01353	27818.2
80	1.212E-14	3.897E-17	1.216E-14	1.01354	41551.6
100	1.256E-15	4.174E-18	1.260E-15	1.01354	81207.7
120	1.245E-16	4.194E-19	1.249E-16	1.01354	140366.8
150	3.682E-18	1.248E-20	3.694E-18	1.01355	274206.2
180	1.047E-19	3.556E-22	1.051E-19	1.01355	473868.3
200	9.72E-21	3.30E-23	9.752E-21	1.01355	650045.4
300	5.65E-26	1.93E-28	5.670E-26	1.01355	2194033.5

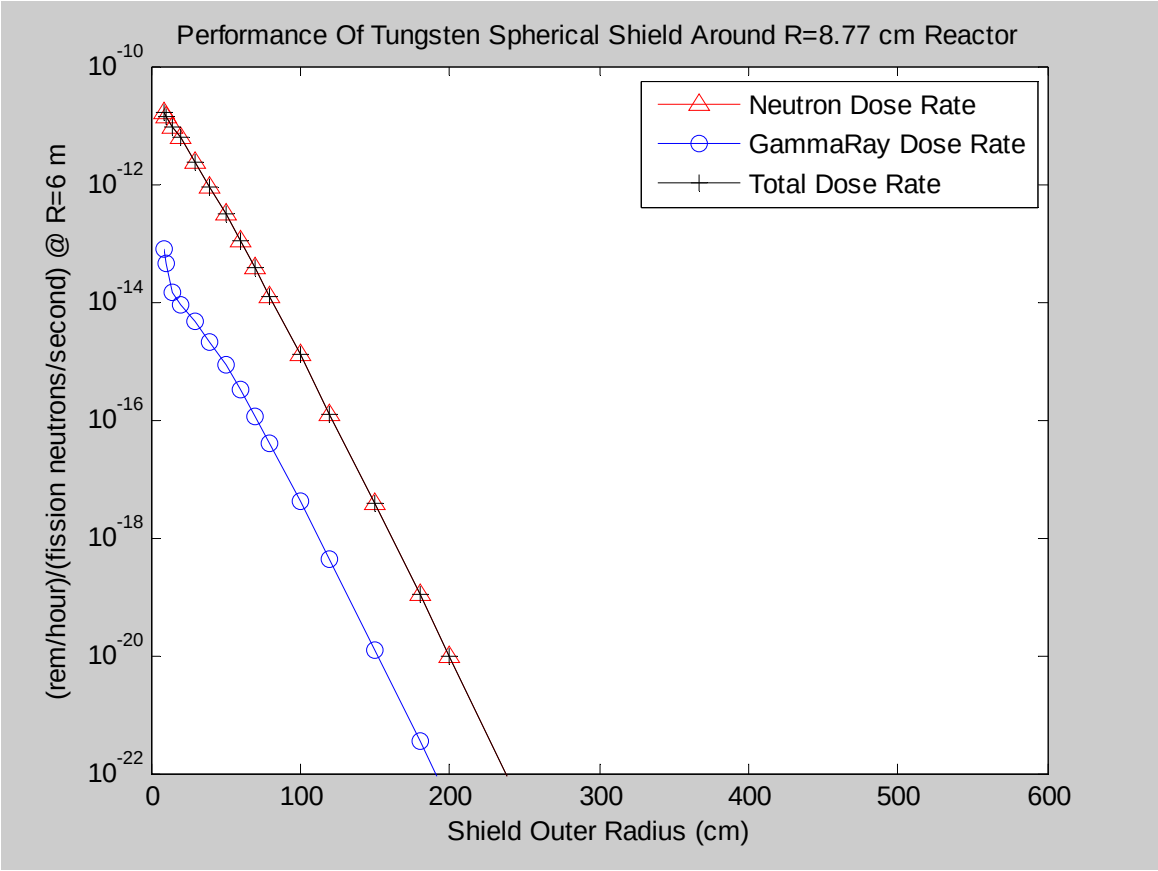


Figure 7-10: Performance of Tungsten Single Material Shield

Table 7-13: Radiation Leakage Vs Size For a Radiation Shield of Uranium-238

Outer Radius of Shell (cm)	neutron dose	gamma dose	total dose	k-effective	shield mass (kg)
8.77	1.563E-11	7.719E-14	1.571E-11	0.72296	0
10	1.422E-11	3.732E-14	1.426E-11	0.79395	26.0
15	9.887E-12	1.348E-14	9.900E-12	0.94394	215.5
20	6.896E-12	8.625E-15	6.905E-12	0.99948	584.5
50	5.295E-13	7.743E-16	5.303E-13	1.03184	9920.7
100	3.214E-15	5.656E-18	3.220E-15	1.03190	79742.6
150	1.483E-17	2.720E-20	1.486E-17	1.03190	269259.2
180	2.362E-18	3.690E-21	2.366E-18	1.03190	465319.1
200	8.26E-19	1.24E-21	8.269E-19	1.03191	638317.8
300	6.10E-19	9.05E-22	6.107E-19	1.03191	2154450.4
400	1.17E-18	1.75E-21	1.175E-18	1.03190	5106919.2
500	6.64E-19	9.94E-22	6.654E-19	1.03189	9974502.9
600	4.31E-19	6.69E-22	4.318E-19	1.03189	17235980.1

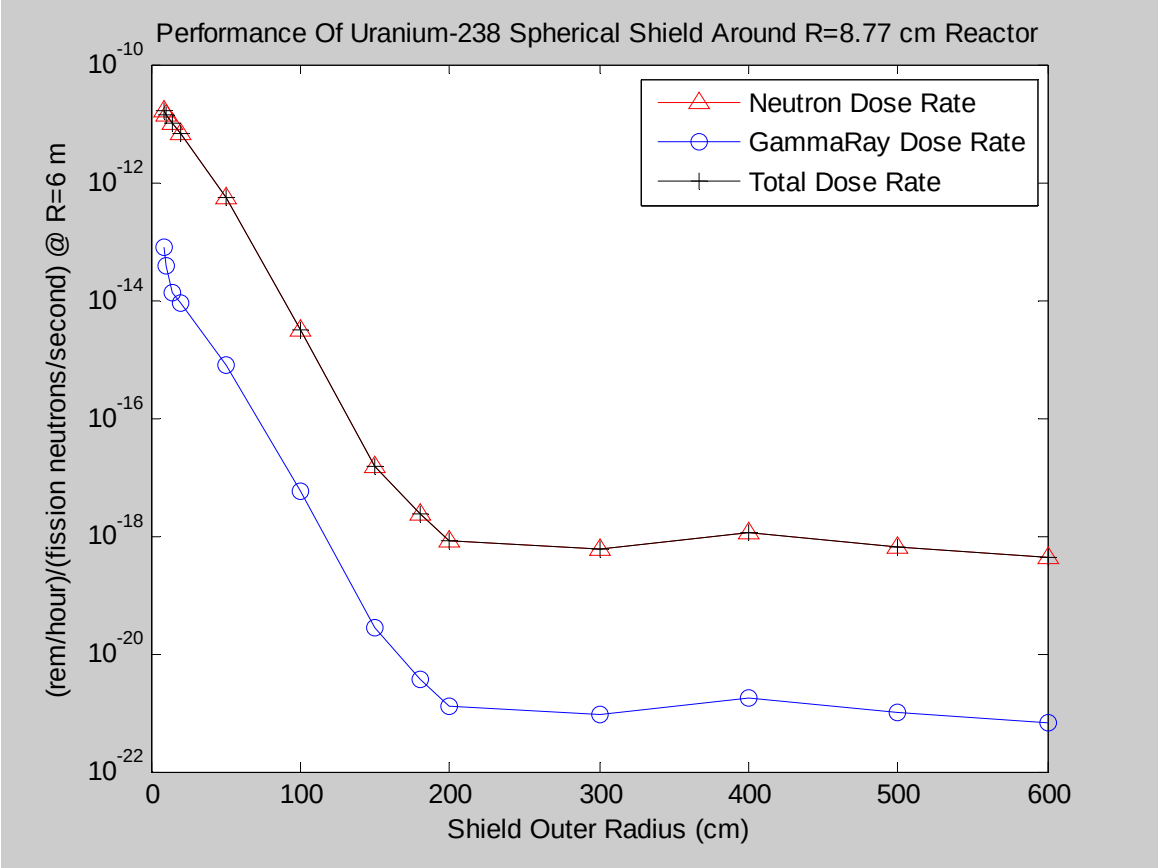


Figure 7-11: Performance of Uranium-238 Single Material Shield

7.4.12 Water (Liquid)

The SAS1X results for a liquid water shield are shown in Table 7-14 and Figure 7-12.

7.5 Comparison of Single Layer Shields Of Different Materials

Figure 7-13 shows on a single graph the neutron attenuation vs. outer radius curves of Figures 7-1 through 7-12 so that performance of the different materials may be directly compared. It shows that Boron and Boron Carbide, both using purified boron-10 isotope, provide the largest neutron attenuation vs. distance. The pure boron has very slightly more neutron attenuation vs. distance than boron carbide, but their curves almost overlay.

For shields whose outer radius is 70 cm or less, plastic is next in neutron attenuation, whereas for shields whose outer radius is 80 cm or more, lithium hydride provides more neutron attenuation for identical radii. For these thick single material shields the natural lithium isotope mix performs slightly better than the lithium-6 version. Curves for the next, water and tungsten cross each other. Uranium follows a little behind tungsten and water until it extends to about two meters, after which it becomes ineffective. Curves for beryllium, iron, graphite, and lead show these materials are not good neutron attenuators.

Figure 7-14 compares the neutron attenuation of these same materials versus shield mass instead of shield outer radius. On this mass basis, LiH using only the lithium-6 isotope provides the most single-material neutron attenuation, followed closely by the natural isotopic mix version of LiH. Polyethylene is next for thinner shields followed by water then boron-10 and boron-10 carbide, while for thicker shields boron-10 is instead next in effectiveness. On this mass basis all the other materials are rather poor neutron attenuators when acting as single material shields.

Figure 7-15 graphs the normalized gamma ray leakage for each of these single material shields versus the shield's outer radius. This is not really the shield's effectiveness in absorbing gamma rays since these shield materials are also generating additional gamma rays due to neutron interactions. The plot shows that the single material shield with the smallest gamma ray leakage is lead for small radii shields and tungsten for larger radii shields, with U-238 almost as effective as tungsten in minimizing gamma rays. Boron carbide and then boron are next, followed by lead and iron. The other materials allow significantly more gamma rays for identical shield outer radii.

Figure 7-16 graphs the same gamma ray leakage versus shield mass for these different single material shields. It shows that for masses below about 4 tonnes, liquid lead provides the lowest gamma ray leakage of the single-material shields considered and that boron and boron carbide have the lowest gamma ray leakage for single material shield masses above 4 tonnes.

The important measure for shield performance is neither the neutron leakage dose nor the gamma ray leakage dose. Instead, the total radiation dose from both neutrons and gammas together is the important measure. That total for these different single-material

Table 7-14: Radiation Leakage Vs Size For a Radiation Shield of Liquid Water

Outer Radius of Shell (cm)	neutron dose	gamma dose	total dose	k-effective	shield mass (kg)
8.77	1.563E-11	7.719E-14	1.571E-11	0.72296	0
10	1.287E-11	9.576E-14	1.297E-11	0.77220	1.4
15	5.817E-12	2.167E-13	6.034E-12	0.91326	11.3
20	2.628E-12	3.053E-13	2.933E-12	0.93915	30.6
30	5.847E-13	3.091E-13	8.938E-13	0.94148	110.1
40	1.524E-13	2.381E-13	3.905E-13	0.94145	264.8
50	4.455E-14	1.728E-13	2.174E-13	0.94143	519.8
60	1.454E-14	1.228E-13	1.373E-13	0.94143	900.3
70	5.211E-15	8.623E-14	9.144E-14	0.94142	1431.3
80	1.997E-15	6.014E-14	6.214E-14	0.94142	2138.0
100	3.250E-16	2.889E-14	2.922E-14	0.94142	4178.4
120	5.519E-17	1.376E-14	1.382E-14	0.94142	7222.4
130	2.275E-17	9.477E-15	9.500E-15	0.94142	9183.4
150	3.835E-18	4.499E-15	4.503E-15	0.94142	14108.9
180	2.600E-19	1.486E-15	1.486E-15	0.94142	24382.2
200	4.325E-20	7.349E-16	7.349E-16	0.94142	33447.2
300	4.675E-24	2.479E-17	2.479E-17	0.94142	112890.9
400	4.631E-28	1.313E-18	1.313E-18	0.94142	267597.2
500	4.446E-32	8.979E-20	8.979E-20	0.94142	522653.5
600	4.694E-36	6.914E-21	6.914E-21	0.94142	903147.3

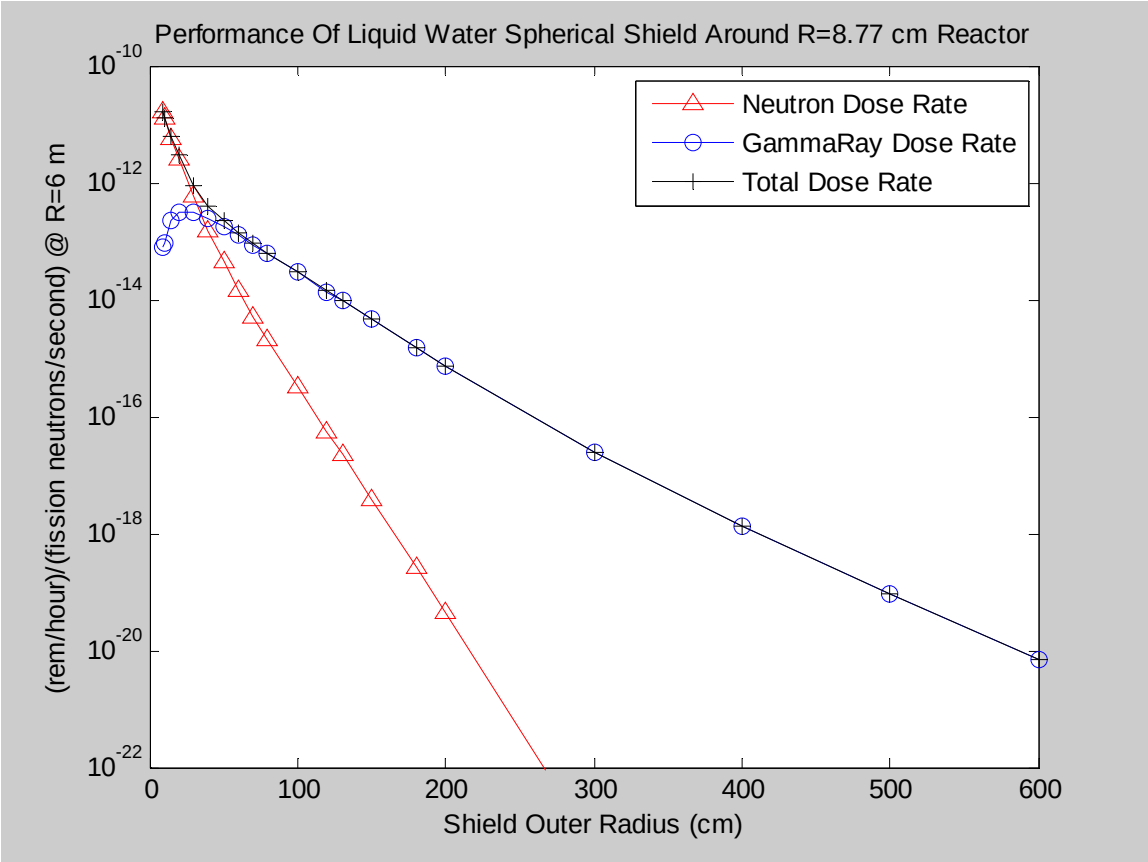


Figure 7-12: Performance of Liquid Water Single Material Shield

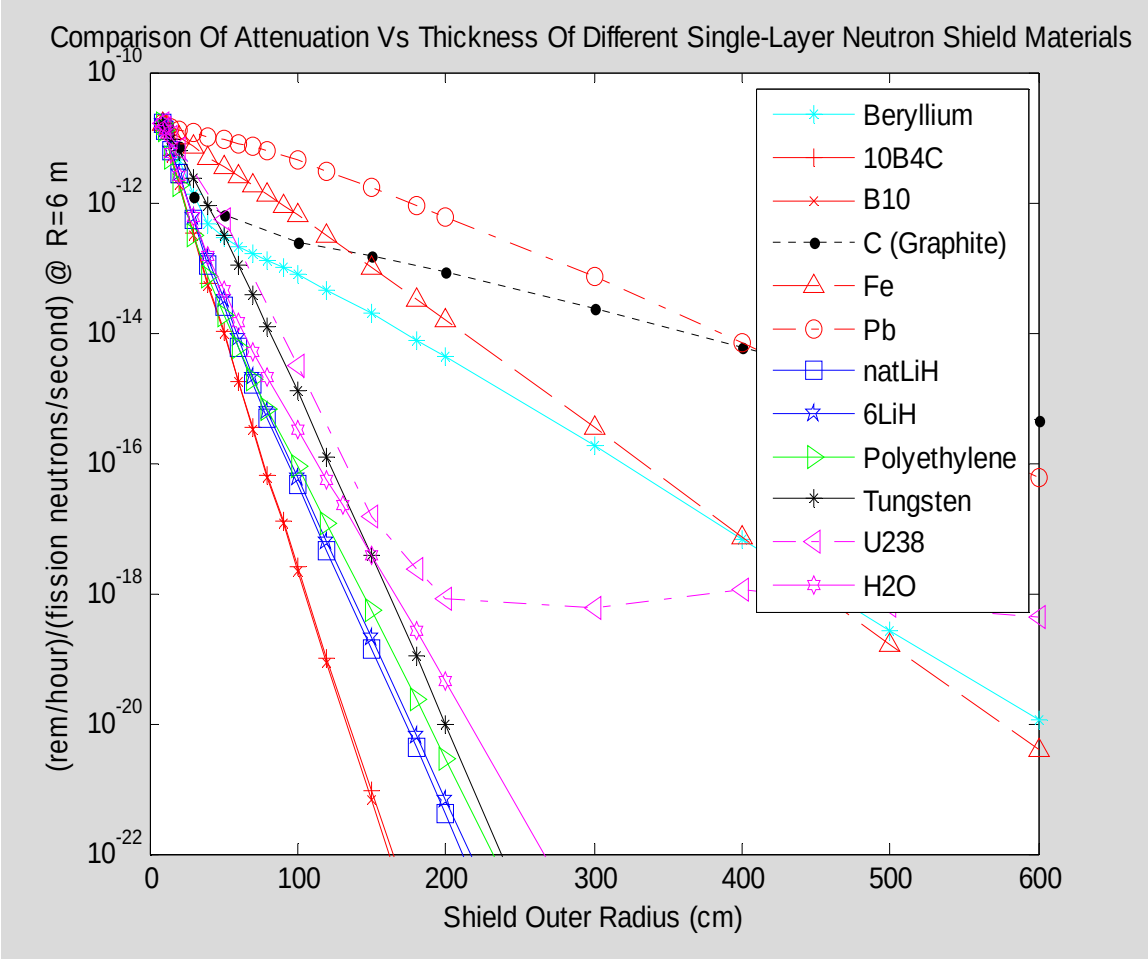


Figure 7-13: Comparison of Neutron Attenuation Vs. Outer Radius of Different Single-Material Spherical Shell Shields

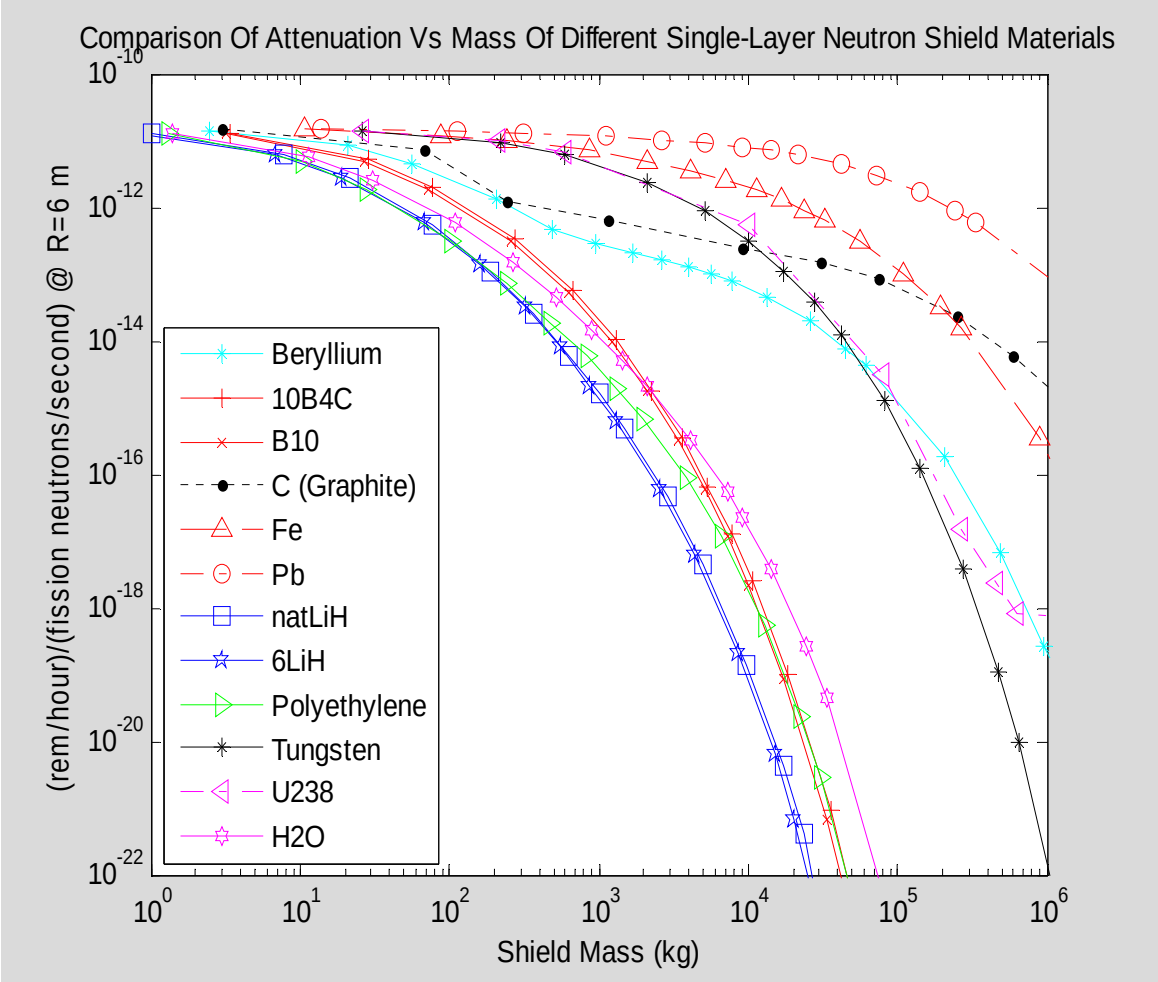


Figure 7-14: Comparison of Neutron Attenuation Vs. Shield Mass of Different Single-Material Spherical Shell Shields

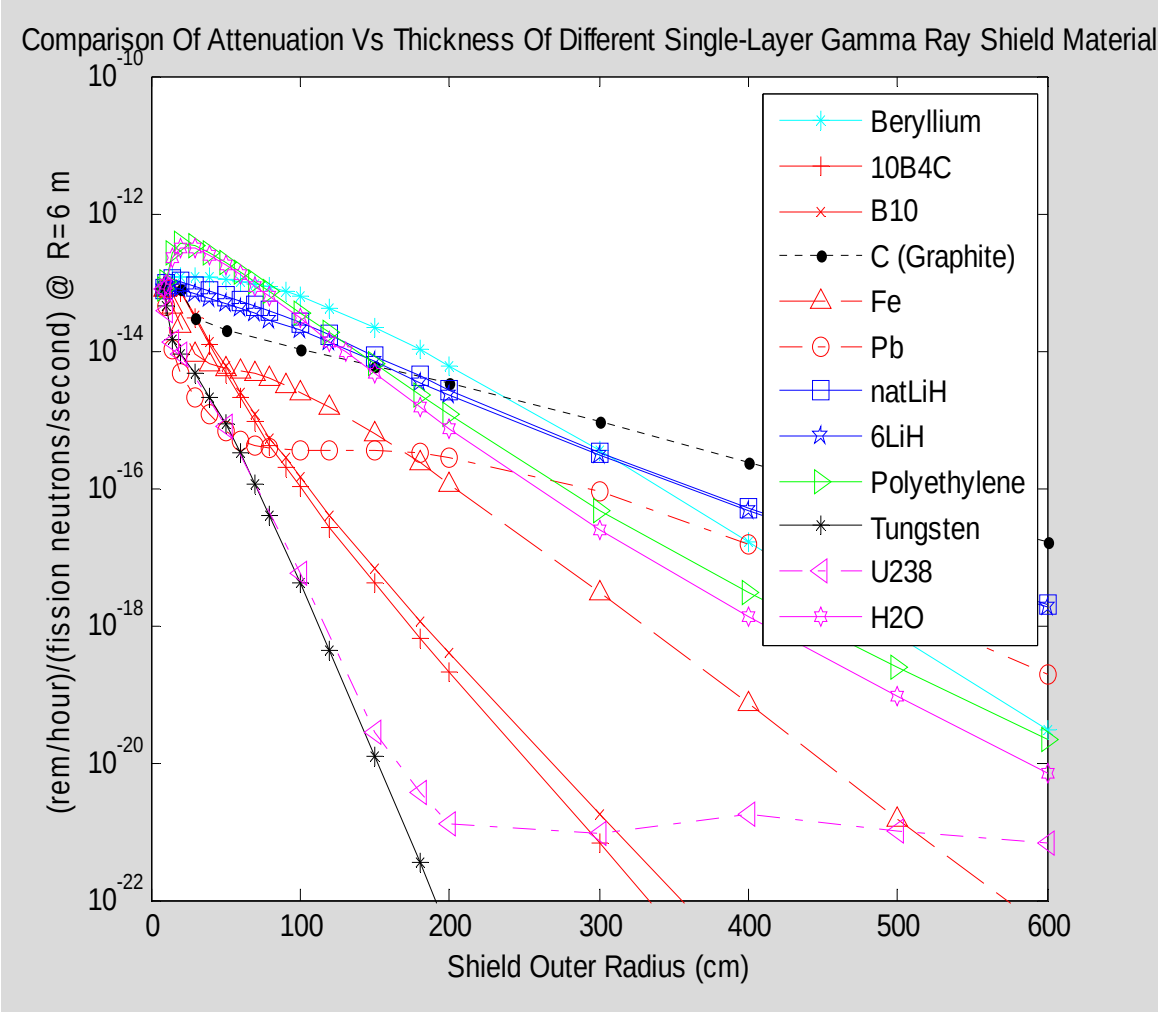


Figure 7-15: Comparison of Gamma Ray Leakage Vs. Shield Outer Radius with Different Single-Material Spherical Shell Shields

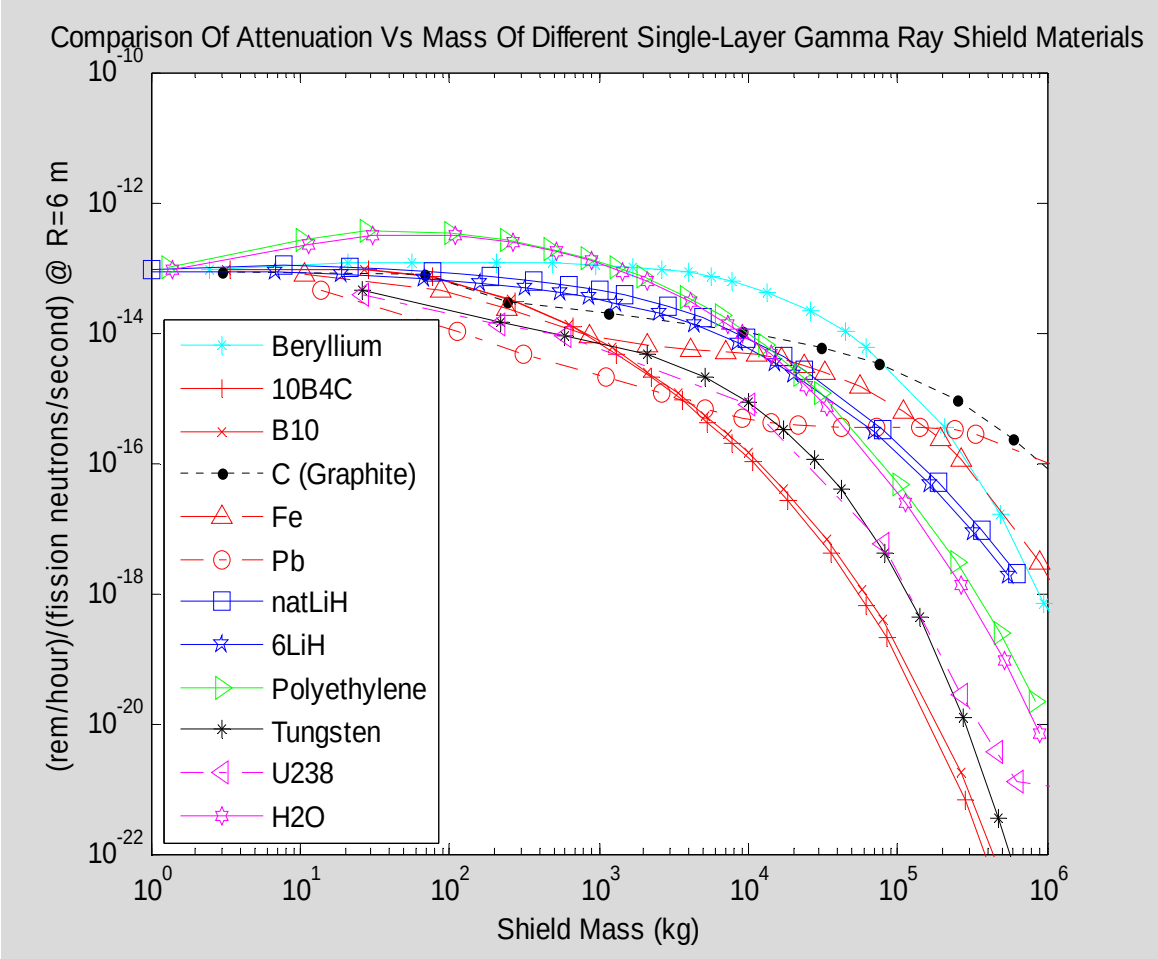


Figure 7-16: Comparison of Gamma Ray Leakage Vs. Shield Mass with Different Single-Material Spherical Shell Shields

shields is plotted versus shield mass in Figure 7-17. **It shows that for single-material shields more massive than one tonne, the single material allowing the least total dose is boron carbide where the boron is completely the boron-10 isotope.** Cubic spline interpolation of the logarithm of the plotted boron carbide total radiation leakage dose data points yields the Table 7-15 shield parameters for the two previously identified limiting radiation dose values 6 meters distant from the reactor center.

Of course, neither 169 tonnes nor 53 tonnes seems light enough for an economical mobile vehicle engine, but they are starting points for studying multiple material shields.

7.6 Multiple Layer Spherical Shell Shields

After several tens of exploratory SAS1X runs modeling different multilayer spherical shell shield configurations, a 4-layer shield configuration was found with a mass of 26.8637 tonnes which reduced the total normalized neutron+gamma dose at r=6 m to 1.0864E-20 (rem/hour)/(fission neutron/second). This level of total radiation leakage is about the same as that of the 169.3374 tonne minimum mass single-material shield, so it was adopted as the starting point for a series of SAS1X runs in a campaign in search of the minimum mass shield with 1.000E-20 leakage. This series of runs is documented in Appendix D, into which the neutron and gamma results from each SAS1X output file have been copied and calculations in preparation for each of the next SAS1X runs in the sequence were implemented in the MSWORD/MATLAB combined form called a MATLAB notebook. During the course of these SAS1X runs it was found that the MATLAB notebook documents failed to transfer MATLAB results correctly back to MSWORD after the document passed a certain number of MATLAB cells. Therefore, it was subdivided into separate MATLAB notebook documents. A total of five were required to cover all these SAS1X runs. These five notebook documents were pasted into a single MSWORD document (without MATLAB) in order to form Appendix D, then 200 pages from their middle were removed for brevity's sake. The remaining Appendix D pages convey the substance of this sequence of runs without all the detail. The final minimum-mass spherical shield configuration found reduced total neutron+gamma radiation leakage 6 m from the reactor's center to 1.000E-20 (rem/hour)/(fission neutron/second), divided as 2.249E-21 from neutrons and 7.751E-21 from gammas. The shield mass was **22.8629 tonnes**.

However, unfortunately the ^{6}LiH densities used in these SAS1X runs and the associated shield mass calculations were slightly in error. The actual density of ^{6}LiH solid at 25 C is 0.68481 g/cm^3 with 0.098268 g/cm^3 contributed by hydrogen and 0.58654 g/cm^3 contributed by lithium-6. Instead, the ^{6}LiH was mistakenly modeled in the Appendix D runs as having a density of 0.7000 g/cm^3 with 0.1000 g/cm^3 from hydrogen and 0.6000g/cm^3 from lithium-6. A SAS1X case keeping the same radial dimensions but

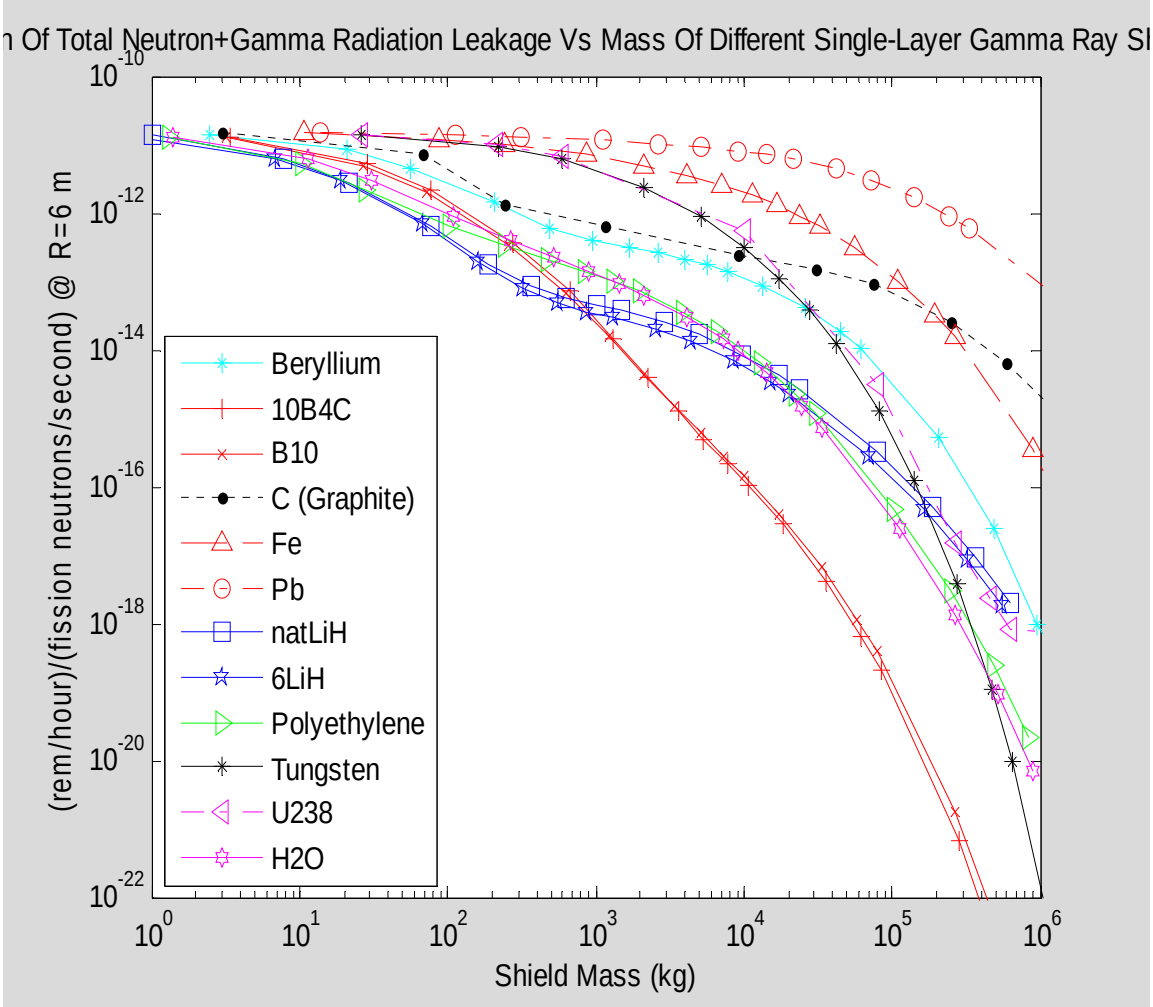


Figure 7-17: Total Leakage Radiation Dose Vs. Shield Mass For Different Single-Material Spherical Shell Shields

Table 7-15: Minimum-Mass Single Material Spherical Shell Shields

Location Of Personnel	**Total Normalized Dose Rate 6 m from Reactor Center**	**Outer Radius Of $^{10}B_4C$ Spherical Shell Shield (cm)**	**Mass Of $^{10}B_4C$ Spherical Shell Shield (kg)**
Outdoors	**1.143E-18**	**170.8**	**52,598.1**
Inside Vehicle	**1.000E-20**	**252.2**	**169,337.4**

with the proper 6LiH densities at 25 C was subsequently run and resulted in the radiation dose rate estimates at r=6 m as follows.

neutrons	2.829E-21
gammas	7.984E-21
total	1.0813E-20

This was 8.1% above the target doserate so further adjustments were then made.

Next, r=138 cm was tried for the outer radius of the 6LiH outer layer.

neutrons	2.072E-21
gammas	7.596E-21
total	0.9668E-20

Next, 137 cm was tried for the outer radius of the 6LiH outer layer.

neutrons	2.366E-21
gammas	7.759E-21
total	1.0125E-20

Next, 137.27 cm was tried for the outer radius of the 6LiH outer layer.

neutrons	2.283E-21
gammas	7.715E-21
total	0.9998E-20

This is close enough to 1E-20, i.e., within 0.02%. . The outer layer's mass was then re-evaluated using this new outer radius and the proper material density in order to reflect this change. Total shield mass increased slightly. The resulting shield parameters are summarized in Table 7-16. Although the Table 7-16 shield is likely close to a minimum-mass design, it is not precisely one. In particular, the above final adjustment to the outer layer's thickness and mass did not consider alternative possible adjustments to inner layers which might perhaps have led to the same radiation leakage result with less of an increase in shield mass. However, the extensive Appendix D series of SAS1X runs did consider such alternatives so the true minimum mass shield with this level of radiation leakage must be not much less than this one's 23 tonne mass.

The normalized maximum dose rate considered for outdoor locations 6 meters from the reactors center, (1.143E-18 rem/hr)/(fission neutron/second), was not the subject of a similar minimum-mass search. Instead, one configuration for it is identified by removing outer layers of the Table 7-16 shield. The result us summarized in Table 7-17.

At the end of these manually guided SAS1X SCALE runs, it was clear that the \process had been extremely time-consuming but it was not clear that the minimum mass design had been found. It had not been feasible to explore possibilities with other materials that have historically been used in shields, simply because of the time involved. For instance, there had not been any investigation of the use of iron in the shielding. This concern motivated the next step, the development of automated methods for shield optimization.

Table 7-16: Spherical Shield For 1.000E-20 (rem/hr)/(fission neutron/sec) Dose Rate @R=6 m

Shell Number	Shell Region Boundaries (cm)	Material Component	Material Component Volume Fraction	Material Component Density (g/cm^3)	Mass (tonnes)
1	8.7731<r<23.9255	Tungsten	1.000	19.3748	1.0567
2	23.9255<r<54.4823	Tungsten	0.36557	19.3748	4.3917
		Boron-10	0.63443	2.37	0.9323
3	54.4823<r<82.7336	Tungsten	0.18365	19.3748	6.0300
		Boron-10	0.81635	2.37	3.2788
4	82.7336<r<84.3561	Lead	0.9997	11.344	1.6140
		Boron-10	0.0003	2.37	0.0001
5	84.3561<r<137.27	^{6}LiH	1.000	0.68481	5.6978
TOTAL SHIELD MASS (tonnes) =					23.0014

Table 7-17: Spherical Shield For 1.143E-18 (rem/hr)/(fission neutron/sec) Dose Rate @R=6 m

Shell Number	Shell Region Boundaries (cm)	Material Component	Material Component Volume Fraction	Material Component Density (g/cm^3)	Mass (tonnes)
1	8.7731<r<23.9255	Tungsten	1.000	19.3748	1.0567
2	23.9255<r<54.4823	Tungsten	0.36557	19.3748	4.3917
		Boron-10	0.63443	2.37	0.9323
3	54.4823<r<82.7336	Tungsten	0.18365	19.3748	6.0300
		Boron-10	0.81635	2.37	3.2788
4	82.7336<r<84.3561	Lead	0.9997	11.344	1.6140
		Boron-10	0.0003	2.37	0.0001
5	84.3561<r<93.775	^{6}LiH	1.000	0.68481	0.6436
TOTAL SHIELD MASS (tonnes) =					17.9472

CHAPTER 8
8 OPTIMAL CONTROL SHIELD DESIGN ALGORITHMS

Any shielding design will permit a fraction of radiation to leak through it. However, leakage can be kept as small as desired by designing sufficiently massive shields of suitable materials. Extremely low radiation leakage limits would force shield designs to have very high mass, perhaps too high for practical surface vehicles on Mars. On the other hand, although high permitted leakage would allow a light-weight shield, it might prove unhealthy. As discussed in Chapter 6, a proper trade-off study of radiation shielding should consider many factors. For scientist-astronauts on an expedition to Mars the existence of many potentially lethal risks not related to radiation exposure should be balanced against the theoretical increase in probability that many years after successfully returning home from Mars they might die early from a fatal cancer. The high radiation background doses from cosmic and solar sources which astronauts expect to absorb should also be considered, since it does not make sense to limit a reactor radiation shield's leakage to a tiny fraction of the irreducible natural background. Since the reactor for a mobile vehicle will not operate at full rated engine power all the time, its likely power profile should be predicted. Human occupancy times for different spatial zones should be estimated along with the likely reactor power levels while occupied.

After choosing requirements for permitted radiation leakage, the next issue is how to provide a shield on Mars meeting those requirements with minimal cost. Each shield design has a total material cost depending on its size, composition, and the unit price values assigned to reflect both production and transportation costs for each of its component materials. Within the set of shield designs meeting a particular specified set of leakage requirements, a design is cost-optimal if there is no other having a lower cost.

The shield design resulting from cost optimization depends on the ratios of assigned material prices per unit mass, not on their absolute values. If these unit prices were all set equal to one then the cost function to minimize would be the total shield mass. This could be appropriate for shields constructed on Earth before deployment to Mars, since the cost per unit mass of interplanetary transportation exceeds raw material costs for most conceivable shielding material candidates by several orders of magnitude. Martian materials added in situ to complete a partial radiation shield deployed from Earth should be assigned a lower unit price since no transportation would be required.

The shield mass-minimization calculations summarized in Chapter 7 were ad hoc, accomplished via sequences of computer runs which required so much human time that only a single design power level was considered and the set of admissible shielding materials was also greatly abbreviated. What is needed instead is an automated way to guide computer runs so that, given radiation leakage requirements, materials, and material unit prices, an optimal minimum cost shield design would be found automatically. **The theory and automation of cost-optimal shield design have been developed within this thesis effort and are described in this chapter.**

Technical references consulted for shielding optimization did not provide mathematical recipes for automated design of cost-optimized shields, so they were developed as part of this work and are presented herein. First there is an original derivation of the Euler-Lagrange equations which must be solved to determine the optimal radiation shield design in the general 3D geometry case. This derivation is followed by the presentation of a class of proposed numerical algorithms to solve them, along with arguments concerning their expected convergence. The 1D spherical geometry version of this algorithm is then discussed in detail, followed by a description of a new SCALE software module developed as part of this work to implement the 1D spherical shield optimization algorithm in conjunction with the existing SCALE system of codes.

8.1 Development of Equations for Cost or Mass Optimized Shields

The present development starts with definitions and notation. It ends with mathematical statements of the function to be minimized and the constraints which must be met.

The differential measure of incremental solid angle for directions on the unit sphere follows a common neutron transport convention [Lewis and Miller 1993, 11]

$$d\Omega \equiv \frac{d\omega}{2\pi}\frac{\sin\theta\, d\theta}{2} \equiv -\frac{d\omega}{2\pi}\frac{d\mu}{2} \quad \text{where } \mu \equiv \cos\theta \tag{8-1}$$

Here, θ and ω respectively represent the polar and meridional angles of the conventional spherical coordinate system, measured with respect to an agreed-upon polar orientation. This convention normalizes the total solid angle on the sphere of directions to unity instead of to the 4π value of the alternative (steradian) solid angle convention, as follows:

$$\oiint d\Omega \equiv \int_0^{2\pi}\frac{d\omega}{2\pi}\int_{-1}^{1}\frac{d\mu}{2} \equiv 1 \tag{8-2}$$

Following established practice the neutron (or gamma ray) radiation's mean density per unit solid angle and energy is modeled by the *angular flux* variable, $\psi\left(\vec{r},\hat{\Omega},E\right)$, a function defined on the six-dimensional space formed as a direct Cartesian product of conventional three dimensional geometric space $\vec{r}\in\Re^3$ with a two dimensional spherical surface defining the unit vector direction of radiation particle motion, $\hat{\Omega}\in\mathbb{S}^2$, and a single nonnegative dimension defining energy per particle, $E\in\Re$. Spatial locations considered in computations are restricted to a bounded convex *spatial domain,* $\mathcal{D}$, a subset of three dimensional space $\vec{r}\in\mathcal{D}\subset\Re^3$. There is also the *flux* variable, $\phi\left(\vec{r},E\right)$, defined as the average over directions of the angular flux variable:

$$\phi\left(\vec{r},E\right) \equiv \oiint d\Omega\,\psi\left(\vec{r},\hat{\Omega},E\right) \tag{8-3}$$

In this development, fission is modeled as an *externally specified* radiation source in order to avoid criticality issues and thus simplify the model's application to shielding

design. In many practical cases the reactor power is feedback controlled to assume a particular steady or load-following power level, so this simplification may be justified.

Nuclear engineering standard practice in evaluating proposed reactor or shielding designs is to use computer codes which extract data from cross section libraries for special calculations. The most precise of these calculations numerically solve the Boltzmann equation for the angular flux function, assuming the stated design configuration and a computational boundary condition.

The steady-state Boltzmann equation for transport of neutral particle radiation is therefore adopted as the fundamental differential equation governing radiation shield designs. It follows:

$$\hat{\Omega} \bullet \nabla \psi\left(\vec{r}, \hat{\Omega}, E\right) + \sigma_t\left(\vec{r}, E\right)\psi\left(\vec{r}, \hat{\Omega}, E\right) = \\ = \oiint d\Omega' \int_0^\infty dE' \sigma_s\left(\vec{r}, \hat{\Omega}' \to \hat{\Omega}, E' \to E\right)\psi\left(\vec{r}, \hat{\Omega}', E'\right) + q\left(\vec{r}, \hat{\Omega}, E\right) \qquad (8\text{-}4)$$

Here, $\sigma_t\left(\vec{r}, E\right)$ is the total energy-dependant macroscopic cross section of the material at location $\vec{r}$, $\sigma_s\left(\vec{r}, \hat{\Omega}' \to \hat{\Omega}, E' \to E\right)$ is the macroscopic differential cross section for scattering at location $\vec{r}$, and $q\left(\vec{r}, \hat{\Omega}, E\right)$ is the radiation source distribution per unit solid angle per unit energy at location $\vec{r}$. Practical radiation sources are frequently isotropic, in which case they may be modeled as $q\left(\vec{r}, E\right)$, independent of the radiation particle direction, $\hat{\Omega}$. In the present development, $q\left(\vec{r}, E\right)$ is zero in purely shielding zones but nonzero within regions representing a fission reactor whose power level is controlled from outside the model. Practical designs for Mars will have a central fission reactor enclosed by shielding, but the mathematical representation is not limited to such configurations.

The Boltzmann equation can be extended to coupled neutron/gamma ray analyses by generalizing its scattering to include "transfer" cross sections, e.g., for (n,γ) processes.

For a convex bounded spatial domain, $\mathcal{D}$, chosen sufficiently large to include the fission reactor radiation source together with any proposed shield design and all important "radiation detector" locations, a "vacuum" boundary condition asserts that no external radiation enters through the domain's boundary, i.e.,

$$\psi\left(\vec{r}, \hat{\Omega}, E\right) = 0, \quad \vec{r} \in \partial\mathcal{D}, \hat{n} \bullet \hat{\Omega} < 0 \qquad (8\text{-}5)$$

where $\hat{n}$ is the outward directed normal unit vector at the spatial domain boundary. This vacuum boundary condition is frequently used in shielding analyses.

The response, $\mathcal{R}$, of a radiation detector located within the domain, $\mathcal{D} \in \Re^3$, is modeled as the inner product between angular flux, $\psi\left(\vec{r},\hat{\Omega},E\right)$ and a detector response function, $R\left(\vec{r},\hat{\Omega},E\right)$, i.e.,

$$\mathcal{R} = \iiint_{\mathcal{D}} dV \int_0^\infty dE \oiint d\Omega \psi\left(\vec{r},\hat{\Omega},E\right) R\left(\vec{r},\hat{\Omega},E\right) dE \,. \tag{8-6}$$

For an omnidirectional small detector the response function can be simplified since it is independent of direction and its spatial extent is small compared to spatial variations in radiation flux. The simplified model involves a delta function of position at the detector's location which drops out during integration.

The model used herein for *D*, the biological radiation dose rate to a human in Sieverts per second, is analogous to a small omnidirectional detector. However, its weighting function *w(E)* converts from flux to dose-rate by taking into account both energy absorption in human body tissue and also quality factors normalizing biological effect to that of gamma rays. Its statement is:

$$\begin{aligned} D = \iiint_{\mathcal{D}} dV \oiint d\Omega \int_0^\infty dE\, \delta\left(\vec{r} - \vec{r}^{\,D}\right) w(E) \psi\left(\vec{r},\hat{\Omega},E\right) = \\ = \int_0^\infty dE w(E) \phi\left(\vec{r}^{\,D},E\right) \end{aligned} \tag{8-7}$$

ANSI standard tabulations of the *w(E)* dose conversion function are available within some cross section libraries.

The calculation of dose rates outside the computational domain in the assumed external vacuum region requires that external angular flux must first be estimated. This requires that the Boltzmann equation solution's computational domain be extended to include each external detector location. However, since q, σ_s, and σ_t are all identically zero throughout the vacuum external region, a simple but exact algorithm based on geometric projection of rays alone can extend the Boltzmann equation solution there.

For the present development, the shield design process consists simply of choosing which materials to use, their quantities, and where to place them. Other important design aspects such as structural or thermal behaviors, manufacturing methods or assembly schemes are ignored here.

It is important for automating the design process that any shield design can be specified by well-defined parameter lists. It is therefore assumed that a sorted complete list of all *m* different acceptable shielding materials exists, along with tabulations of their material and nuclear properties, indexed here by $i \in I \equiv \{1,2,3,...m\}$. The mass density of a pure sample of material number *i* is denoted by ρ_i, its price per unit mass by p_i, and its cost per unit volume by $c_i = p_i \rho_i$. The total macroscopic cross section of material number *i* is

denoted by $\sigma_t^{(i)}(E)$, and the macroscopic differential scattering cross section of material number *i* is $\sigma_s^{(i)}\left(\hat{\Omega}' \to \hat{\Omega}, E' \to E\right)$.

The radiation shield may include mixtures of the *m* acceptable materials using any possible set of mixing volume fractions. Therefore, the specification of how the volume fraction of each material varies over the spatial domain completely defines the shield design. Since these volume fraction functions are the control variables which are controlled by a shield designer, modeling terminology is adopted from the control theory field. The volume fraction for material number *i* is denoted here by the scalar function of position, $u_i(\vec{r})$, and this notation is used for each of the *m* materials. The range of possible volume fractions for each material is bounded at any location to be a nonnegative value which also cannot exceed one, as expressed by the following inequalities:

$$0 \le u_i(\vec{r}), \qquad i \in I \tag{8-8}$$

$$u_i(\vec{r}) \le 1, \quad i \in I \tag{8-9}$$

In addition to the 2m inequalities of Eqs. 8-8 and 8-9, there is also another inequality linking all the materials together through the assertion that the total sum of their volume fractions cannot exceed one:

$$\sum_{i=1}^{m} u_i(\vec{r}) \le 1 \tag{8-10}$$

It should be noted that Eqs 8-8 and 8-10 together imply Eq.8-9, so actually there are only (m+1) *independent* inequality constraints..

Employing these definitions of the shield design's scalar control variables, $u_i(\vec{r})$, allows terms in Eq. 8-4 to be rewritten using summations, as follows:

$$\sigma_t(\vec{r}, E) = \sum_{i=1}^{m} \sigma_t^{(i)}(E) u_i(\vec{r}) \tag{8-11}$$

$$\sigma_s\left(\vec{r}, \hat{\Omega}' \to \hat{\Omega}, E' \to E\right) = \sum_{i=1}^{m} \sigma_s^{(i)}\left(\hat{\Omega}' \to \hat{\Omega}, E' \to E\right) u_i(\vec{r}) \tag{8-12}$$

Similarly, the shield's mass density function becomes the following:

$$\rho(\vec{r}) = \sum_{i=1}^{m} \rho_i u_i(\vec{r}) \tag{8-13}$$

and its cost per unit volume becomes the following:

$$c(\vec{r}) = \sum_{i=1}^{m} p_i \rho_i u_i(\vec{r}) = \sum_{i=1}^{m} c_i u_i(\vec{r}) \tag{8-14}$$

A more compact notation imported from control theory defines an m-component control vector of functions:

$$\underline{u}(\vec{r}) \equiv \begin{pmatrix} u_1(\vec{r}) \\ u_2(\vec{r}) \\ \vdots \\ u_m(\vec{r}) \end{pmatrix} \tag{8-15}$$

Using this notation, defining a radiation shield designed using up to m different types of materials is completely equivalent to specifying this m-vector function of position, $\underline{u}(\vec{r})$. Following Pontryagin's notation [Pontryagin et al 1962] the closed, convex region in m-dimensional control space defined by Eqs. 8-8 and 8-10 is the *admissible control set*, *U*, and their full set of inequality constraints is denoted by the simple shorthand statement,

$$\underline{u} \in U \tag{8-16}$$

The following m-component vectors are defined. Note that they are independent of position:

$$\underline{\sigma}_t(E) \equiv \begin{pmatrix} \sigma_t^{(1)}(E) \\ \sigma_t^{(2)}(E) \\ \vdots \\ \sigma_t^{(m)}(E) \end{pmatrix} \tag{8-17}$$

$$\underline{\sigma}_s\left(\hat{\Omega}' \to \hat{\Omega}, E' \to E\right) \equiv \begin{pmatrix} \sigma_s^{(1)}\left(\hat{\Omega}' \to \hat{\Omega}, E' \to E\right) \\ \sigma_s^{(2)}\left(\hat{\Omega}' \to \hat{\Omega}, E' \to E\right) \\ \vdots \\ \sigma_s^{(m)}\left(\hat{\Omega}' \to \hat{\Omega}, E' \to E\right) \end{pmatrix} \tag{8-18}$$

$$\underline{\rho} = \begin{pmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_m \end{pmatrix} \tag{8-19}$$

$$\underline{p} = \begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_m \end{pmatrix} \tag{8-20}$$

$$\underline{c} = \begin{pmatrix} p_1\rho_1 \\ p_2\rho_2 \\ \vdots \\ p_m\rho_m \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{pmatrix} \tag{8-21}$$

The alternative use of symbols ρ and c with neither subscript nor underscore to signify local mass or cost densities in Eqs 8-13 and 8-14 should not cause confusion.

Using this notation, Eqs 8-11 through 8-14 are compactly rewritten as follows:

$$\sigma_t(\vec{r},E)=\underline{\sigma}_t^{\,T}(E)\underline{u}(\vec{r}) \tag{8-22}$$

$$\sigma_s\left(\vec{r},\hat{\Omega}'\to\hat{\Omega},E'\to E\right)=\underline{\sigma}_s^{\,T}\left(\hat{\Omega}'\to\hat{\Omega},E'\to E\right)\underline{u}(\vec{r}) \tag{8-23}$$

$$\rho(\vec{r})=\underline{\rho}^{\,T}\,\underline{u}(\vec{r}) \tag{8-24}$$

$$c(\vec{r})=\underline{c}^{\,T}\,\underline{u}(\vec{r}) \tag{8-25}$$

where the superscript *T* signifies the vector transpose. These control vector notations can be used to clearly separate out materials properties from the control function $\underline{u}(\vec{r})$ which compactly specifies a design. In this notation the shield's total cost, which the optimal design process must minimize subject to specified radiation leakage constraints, is expressed as follows:

$$C=\iiint_{\mathcal{D}} dV\,c(\vec{r})=\iiint_{\mathcal{D}}\underline{c}^{\,T}\,\underline{u}(\vec{r})dV=\underline{c}^{\,T}\iiint_{\mathcal{D}} dV\,\underline{u}(\vec{r}) \tag{8-26}$$

Note that in the special case that material unit prices are all set to one this total cost equals the total shield mass:

$$M=\iiint_{\mathcal{D}} dV\rho(\vec{r})=\iiint_{\mathcal{D}}\underline{\rho}^{\,T}\,\underline{u}(\vec{r})dV=\underline{\rho}^{\,T}\iiint_{\mathcal{D}}\underline{u}(\vec{r})dV \tag{8-27}$$

The Boltzmann equation with an external vacuum boundary condition is rewritten using this notation as follows:

$$\begin{aligned}&\hat{\Omega}\bullet\nabla\psi\left(\vec{r},\hat{\Omega},E\right)+\underline{\sigma}_t^{\,T}(E)\underline{u}(\vec{r})\psi\left(\vec{r},\hat{\Omega},E\right)\\&-\oint\!\!\oint d\Omega'\int_0^\infty dE'\,\underline{\sigma}_s^{\,T}\left(\hat{\Omega}'\to\hat{\Omega},E'\to E\right)\underline{u}(\vec{r})\psi\left(\vec{r},\hat{\Omega}',E'\right)-q\left(\vec{r},\hat{\Omega},E\right)=0\\&\text{for}\quad\vec{r}\in\mathcal{D},\underline{u}\in U;\\&\text{with vacuum boundary condition:}\quad\psi\left(\vec{r},\hat{\Omega},E\right)\Big|_{\substack{\vec{r}\in\partial\mathcal{D}\\ \hat{n}\bullet\hat{\Omega}<0}}=0\end{aligned} \tag{8-28}$$

The full radiation shielding design problem statement must include not only a specification of the fission source, $q(\vec{r},E)$, but also a statement of the permitted radiation leakage through the shield as defined by allowable dose rates at a specified set of "detector" radiation constraint locations. Radiation shield designs typically need to limit radiation at more than a single location. To emphasize the possible use of alternative specification schemes to denote radiation constraint locations, a general abstract notation is adopted. The "α" index from the index set *A* appearing below in Eq.8-29 could represent an integer for a finite set or a countably infinite set of $\vec{r}_\alpha^{\,D}$ locations, or alternatively, it could represent a 2-element vector of real numbers adequate to uniquely

specify any point location on a continuous surface. It could even represent a 3-element vector of real numbers, allowing maximum radiation dose limits to be specified everywhere throughout a volume. The statement of permitted radiation leakage is thus written as follows:

$$D_\alpha \le D_\alpha^{MAX} \quad \forall\, \alpha \in A \tag{8-29a}$$

where

$$D_\alpha \equiv \iiint_{\mathcal{D}} dV \oiint d\Omega \int_0^\infty dE\, \delta\left(\vec{r} - \vec{r}_\alpha^{\,D}\right) w(E) \psi\left(\vec{r}, \hat{\Omega}, E\right) \equiv \int_0^\infty w(E) \phi\left(\vec{r}_\alpha^{\,D}, E\right) dE \tag{8-29b}$$

This completes the mathematical development of the shield design optimization problem to be solved. For convenience, it is restated in summary form in Table 8-1.

A cautionary remark is in order at this point. Table 8-1 defers specifying smoothness requirements for $\underline{u}(\vec{r})$. Discontinuous $\underline{u}(\vec{r})$ may allow locally different optimal shield designs to have identical attenuation and cost, whereas limiting $\underline{u}(\vec{r})$ to continuous functions may guarantee the uniqueness of $\underline{u}(\vec{r})$ optimal shield design solutions.

The shield cost optimization design problem is to choose $\underline{u}(\vec{r}) \in U; \quad \vec{r} \in \mathcal{D}$ so that the functional *C* as given by Eq. 8-26 is minimized while Eqs 8-28 and 8-29 are both satisfied.

$$C = \underline{c}^T \iiint_{\mathcal{D}} \underline{u}(\vec{r}) dV \tag{8-26}$$

$$\hat{\Omega} \bullet \nabla \psi\left(\vec{r}, \hat{\Omega}, E\right) + \underline{\sigma}_t^{\,T}(E) \underline{u}(\vec{r}) \psi\left(\vec{r}, \hat{\Omega}, E\right)$$
$$- \oiint d\Omega' \int_0^\infty dE' \underline{\sigma}_s^{\,T}\left(\hat{\Omega}' \to \hat{\Omega}, E' \to E\right) \underline{u}(\vec{r}) \psi\left(\vec{r}, \hat{\Omega}', E'\right) - q\left(\vec{r}, \hat{\Omega}, E\right) = 0 \tag{8-28}$$

for $\vec{r} \in \mathcal{D}, \underline{u} \in U;$

with vacuum boundary condition: $\psi\left(\vec{r}, \hat{\Omega}, E\right)\Big|_{\substack{\vec{r} \in \partial\mathcal{D} \\ \hat{n} \bullet \hat{\Omega} < 0}} = 0$

$$D_\alpha \le D_\alpha^{MAX} \quad \forall\, \alpha \in A \tag{8-29a}$$

where

$$D_\alpha \equiv \iiint_{\mathcal{D}} dV \oiint d\Omega \int_0^\infty dE\, \delta\left(\vec{r} - \vec{r}_\alpha^{\,D}\right) w(E) \psi\left(\vec{r}, \hat{\Omega}, E\right) \equiv \int_0^\infty w(E) \phi\left(\vec{r}_\alpha^{\,D}, E\right) dE \tag{8-29b}$$

Figure 8-1: Mathematical Statement of Shield Design Optimization Problem

Remarks: A shield design optimization problem with no solution occurs if there is not enough distance between the radiation source and the doserate constraint location to provide required attenuation of the radiation using the defined set of shielding materials. A shield design optimization problem requiring no shield whatsoever, occurs if the distance alone between the source and the doserate constraint location provides enough attenuation. In all other problems, at least one doserate constraint is satisfied as an equality by the solution, i.e., it is effective.

8.1.1 Development of Euler-Lagrange Equations

This development proceeds by next combining Equations 8-26, 8-28, and 8-29 into a single equation. Since Equation 8-28 is an equality constraint, Lagrange's method of undetermined multipliers is used to adjoin it to Equation 8-26. Since the terms in Equation 8-28 are all functions of $\left(\vec{r},\hat{\Omega}',E'\right)$, the Lagrange multiplier function introduced here, $\lambda \equiv \lambda\left(\vec{r},\hat{\Omega},E\right)$, is used to form the following inner product (I.P.).

$$I.P. = \iiint_{\mathcal{D}} dV \oiint d\Omega \int_0^\infty dE \left[\lambda\left(\vec{r},\hat{\Omega},E\right) \left[\begin{array}{l} -\underline{\sigma}_t{}^T(E)\psi\left(\vec{r},\hat{\Omega},E\right)+ \\ \left(\oiint d\Omega' \int_0^\infty dE' \underline{\sigma}_s{}^T\left(\hat{\Omega}'\to\hat{\Omega},E'\to E\right)\psi\left(\vec{r},\hat{\Omega}',E'\right) \right) \\ +q\left(\vec{r},\hat{\Omega},E\right)-\hat{\Omega}\bullet\nabla\psi\left(\vec{r},\hat{\Omega},E\right) \end{array} \right] \underline{u}\left(\vec{r}\right) \right] \tag{8-30}$$

The units of λ must convert to cost units to be compatible with Eq. .8-26. However, it is clear by the construction of Eq. 8-30 that this inner product is identically zero, numerically.

Equation 8.29 is combined with Eq., 8-26 by using Kuhn-Tucker multipliers, a special form of Lagrange multipliers for nonlinear optimization with inequality constraints. The Kuhn-Tucker theorem, developed for the mathematical field of nonlinear programming, defines globally optimal corner condition solutions. Each Kuhn-Tucker multiplier, τ_α, is a nonnegative scalar value whose positivity must be coordinated with a nonpositive constraint function, $\left(D_\alpha - D_\alpha^{MAX}\right)$, so that their product, $\tau_\alpha\left(D_\alpha - D_\alpha^{MAX}\right)$, is zero but its derivative with respect to D_α is nonnegative. Thus, the undetermined multipliers, τ_α, are limited as follows:

$$\tau_\alpha \begin{cases} = 0 \text{ if } D_\alpha - D_\alpha^{MAX} < 0 \\ \geq 0 \text{ if } D_\alpha - D_\alpha^{MAX} = 0 \end{cases}$$

where

$$D_\alpha \equiv \int_0^\infty w(E)\phi\left(\vec{r}_\alpha^{\,D}, E\right)dE \tag{8-31}$$

$$\equiv \iiint_{\mathcal{D}} dV\,\delta\left(\vec{r} - \vec{r}_\alpha^{\,D}\right) \oiint d\Omega \int_0^\infty dE\, w(E)\psi\left(\vec{r}, \hat{\Omega}, E\right)$$

The τ_α vary as a function only of the location indexing variable, $\alpha \in A$. Their units must convert from biological dose rates to cost units in order to be compatible with Eq. 8-26.

Kuhn-Tucker constraints which are exactly met as equalities in the optimized solution are identified in publications as *effective* constraints, sometimes alternatively called *active* or *binding* constraints. These do influence optimal design solutions. The other constraints, which are met as strict inequalities and whose multiplier values τ_α are therefore zero according to Eq. 8-31, are termed *ineffective, inactive, or slack* constraints. They do not influence the optimal design solution.

With multipliers affixed, the augmented version of Equation 8-26 follows:

$$C = \underline{c}^T \iiint_{\mathcal{D}} dV\,\underline{u}(\vec{r}) +$$

$$\iiint_{\mathcal{D}} dV \oiint d\Omega \int_0^\infty dE \lambda\left(\vec{r}, \hat{\Omega}, E\right) \left[\begin{array}{l} \left(\begin{array}{l} -\underline{\sigma}_t{}^T(E)\psi\left(\vec{r}, \hat{\Omega}, E\right) + \\ \oiint d\Omega' \int_0^\infty dE' \underline{\sigma}_s{}^T\left(\hat{\Omega}' \to \hat{\Omega}, E' \to E\right)\psi\left(\vec{r}, \hat{\Omega}', E'\right) \end{array} \right) \underline{u}(\vec{r}) \\ + q\left(\vec{r}, \hat{\Omega}, E\right) - \hat{\Omega} \bullet \nabla\psi\left(\vec{r}, \hat{\Omega}, E\right) \end{array} \right]$$

$$+ \int_{\alpha \in A} d\mu_\alpha \tau_\alpha \left(D_\alpha - D_\alpha^{MAX}\right) \tag{8-32}$$

In Eq. 8-32, the symbol, $\int_{\alpha \in A} d\mu_\alpha(\ )$ denotes abstract measure-space integration with respect to the (unspecified) set measure, μ_α, over the doserate constraint location index set, A. Abstract integration notation can signify a discrete sum if μ_α is a Dirac measure defined on a countable index set or a more conventional integration if μ_α is, e.g., a Lebesgue or Riemann measure.

If the specified D_α^{MAX} is stated as an interpolation function of position, $D_\alpha^{MAX}(\vec{r})$, having enough smoothness to be sampled by Dirac delta functions, then Eq. 8-32 can be rewritten in the following equivalent form:

$$C = \iiint_{\mathcal{D}} dV \left[\begin{array}{l} \underline{c}^T \underline{u}(\vec{r}) - \int_{\alpha \in A} \tau_\alpha \delta(\vec{r} - \vec{r}_\alpha^{\,D}) D_\alpha^{MAX}(\vec{r}) d\mu_\alpha + \\ \oiint d\Omega \int_0^\infty dE \left[\begin{array}{l} \lambda(\vec{r}, \hat{\Omega}, E) \left[\begin{array}{l} \left(\begin{array}{l} -\underline{\sigma}_t^{\,T}(E) \psi(\vec{r}, \hat{\Omega}, E) + \\ \oiint d\Omega' \int_0^\infty dE' \underline{\sigma}_s^{\,T}(\hat{\Omega}' \to \hat{\Omega}, E' \to E) \psi(\vec{r}, \hat{\Omega}', E') \end{array} \right) \underline{u}(\vec{r}) \\ + q(\vec{r}, \hat{\Omega}, E) - \hat{\Omega} \bullet \nabla \psi(\vec{r}, \hat{\Omega}, E) \end{array} \right] \\ + \int_{\alpha \in A} \tau_\alpha \delta(\vec{r} - \vec{r}_\alpha^{\,D}) w(E) \psi(\vec{r}, \hat{\Omega}, E) d\mu_\alpha \end{array} \right] \end{array} \right] \tag{8-33}$$

It is customary to simplify notation by defining a scalar *Hamiltonian* function:

$$H(\lambda, \tau_\alpha, \psi, \underline{u}, \vec{r}) \equiv \underline{c}^T \underline{u}(\vec{r}) - \int_{\alpha \in A} \tau_\alpha \delta(\vec{r} - \vec{r}_\alpha^{\,D}) D_\alpha^{MAX}(\vec{r}) d\mu_\alpha$$
$$+ \oiint d\Omega \int_0^\infty dE \left\{ \begin{array}{l} \lambda(\vec{r}, \hat{\Omega}, E) \left[\begin{array}{l} \left(\begin{array}{l} -\underline{\sigma}_t^{\,T}(E) \psi(\vec{r}, \hat{\Omega}, E) + \\ \oiint d\Omega' \int_0^\infty dE' \underline{\sigma}_s^{\,T}(\hat{\Omega}' \to \hat{\Omega}, E' \to E) \psi(\vec{r}, \hat{\Omega}', E') \end{array} \right) \underline{u}(\vec{r}) \\ + q(\vec{r}, \hat{\Omega}, E) - \hat{\Omega} \bullet \nabla \psi(\vec{r}, \hat{\Omega}, E) \end{array} \right] \\ + \int_{\alpha \in A} \tau_\alpha \delta(\vec{r} - \vec{r}_\alpha^{\,D}) w(E) \psi(\vec{r}, \hat{\Omega}, E) d\mu_\alpha \end{array} \right\} \tag{8-34}$$

so that Eq. 8-33 can be rewritten as

$$C = \iiint_{\mathcal{D}} dV \left[H(\lambda, \tau_\alpha, \psi, \underline{u}, \vec{r}) - \int_0^\infty dE \oiint d\Omega \, \hat{\Omega} \bullet \lambda \nabla \psi \right] \tag{8-35}$$

This result is next rewritten in an equivalent form by using a sequence of mathematical identities to integrate its last term by parts in three dimensions. The integration sequence is first swapped.

$$\iiint_{\mathcal{D}} dV \int_0^\infty dE \oiint d\Omega \left(\hat{\Omega} \bullet \lambda \nabla \psi \right) \equiv \oiint d\Omega \, \hat{\Omega} \bullet \int_0^\infty dE \iiint_{\mathcal{D}} dV (\lambda \nabla \psi)$$

Then a corollary of Gauss' divergence theorem is invoked, i.e.,

$$\iiint_{\mathcal{D}} dV \left(\nabla f(\vec{r}) \right) \equiv \iint_{\partial \mathcal{D}} d\sigma \hat{n} f(\vec{r})$$

where $\hat{n}$ is the outward-directed normal unit vector on the spatial domain's boundary, $\partial \mathcal{D}$, where dσ is a differential area on that boundary, and *f* is any scalar function of

position sufficiently smooth so that its gradient is well-defined. Making use also of the product rule identity for gradients,

$$\nabla(gh) \equiv g\nabla(h) + h\nabla(g)$$

yields an integration by parts identity for any fixed value of energy, E, and for any fixed propagation direction, $\hat{\Omega}$:

$$\iiint_{\mathcal{D}} dV\, \lambda\left(\vec{r},\hat{\Omega},E\right)\nabla\psi\left(\vec{r},\hat{\Omega},E\right) =$$

$$= \iint_{\partial\mathcal{D}} d\sigma \hat{n}\,\psi\left(\vec{r},\hat{\Omega},E\right)\lambda\left(\vec{r},\hat{\Omega},E\right) - \iiint_{\mathcal{D}} dV\,\psi\left(\vec{r},\hat{\Omega},E\right)\nabla\lambda\left(\vec{r},\hat{\Omega},E\right)$$

Substituting this integration by parts identity into Eq. 8-35 results in the following equivalent form:

$$C = -\oint\!\!\oint d\Omega\hat{\Omega}\bullet\int_0^\infty dE\iint_{\partial\mathcal{D}} d\sigma\hat{n}\,\psi\left(\vec{r},\hat{\Omega},E\right)\lambda\left(\vec{r},\hat{\Omega},E\right) +$$

$$\iiint_{\mathcal{D}} dV\left[H\left(\lambda,\nu_\alpha,\psi,\underline{u},\vec{r}\right) + \oint\!\!\oint d\Omega\hat{\Omega}\bullet\int_0^\infty dE\,\psi\left(\vec{r},\hat{\Omega},E\right)\nabla\lambda\left(\vec{r},\hat{\Omega},E\right)\right] \tag{8-36}$$

So far, the only specified constraint on the Lagrange multiplier function $\lambda\left(\vec{r},\hat{\Omega},E\right)$ is that it must be sufficiently smooth so that its gradient is everywhere well-defined. However, to further simplify Eq.8-36 it is specified at this point in the development that $\lambda\left(\vec{r},\hat{\Omega},E\right)$ is zero for all outward directions at all locations on the domain boundary, i.e.,

$$\lambda\left(\vec{r},\hat{\Omega},E\right)\Big|_{\substack{\hat{n}\bullet\hat{\Omega}>0 \\ \vec{r}\in\partial\mathcal{D}}} = 0 \tag{8-37}$$

Since the vacuum boundary condition of Eq.8-5 already specifies the angular flux to be zero for all inward directions on the domain boundary, the two constraints together guarantee that the boundary integral of the product of the functions is zero, i.e.,

$$\iint_{\partial\mathcal{D}} d\sigma\hat{n}\,\psi\left(\vec{r},\hat{\Omega},E\right)\lambda\left(\vec{r},\hat{\Omega},E\right) = 0 \tag{8-38}$$

Thus, the first term in Eq. 8-36 is zero, so the total cost function can be rewritten as follows:

$$C = \iiint_{\mathcal{D}} dV\left[H\left(\lambda,\tau_\alpha,\psi,\underline{u},\vec{r}\right) + \oint\!\!\oint d\Omega\hat{\Omega}\bullet\int_0^\infty dE\,\psi\left(\vec{r},\hat{\Omega},E\right)\nabla\lambda\left(\vec{r},\hat{\Omega},E\right)\right] \tag{8-39}$$

Next, the Hamiltonian is put into an equivalent form in which the angular flux term is more accessible for algebraic manipulation. Swapping the dummy variables of integration,

$$\iint_{\Omega} d\Omega \int_0^{\infty} dE \lambda(\vec{r},\hat{\Omega},E) \iint_{\Omega} d\Omega' \int_0^{\infty} dE' \underline{\sigma}_s{}^T(\hat{\Omega}' \to \hat{\Omega}, E' \to E) \psi(\vec{r},\hat{\Omega}',E') \equiv$$
$$= \iint_{\Omega} d\Omega \int_0^{\infty} dE \psi(\vec{r},\hat{\Omega},E) \iint_{\Omega} d\Omega' \int_0^{\infty} dE' \underline{\sigma}_s{}^T(\hat{\Omega} \to \hat{\Omega}', E \to E') \lambda(\vec{r},\hat{\Omega}',E') \tag{8-40}$$

With this substitution, the Hamiltonian is rewritten equivalently as follows:

$$H(\lambda,\tau_\alpha,\psi,\underline{u},\vec{r}) \equiv \underline{c}^T \underline{u}(\vec{r}) - \int_{\alpha \in A} \tau_\alpha \delta(\vec{r} - \vec{r}_\alpha^{\,D}) D_\alpha^{MAX}(\vec{r}) d\mu_\alpha$$
$$+ \iint_{\Omega} d\Omega \int_0^{\infty} dE \begin{bmatrix} \psi(\vec{r},\hat{\Omega},E) \begin{bmatrix} \left(\begin{array}{l} -\underline{\sigma}_t{}^T \lambda(\vec{r},\hat{\Omega},E) \\ + \iint_{\Omega} d\Omega' \int_0^{\infty} dE' \underline{\sigma}_s{}^T(\hat{\Omega} \to \hat{\Omega}', E \to E') \lambda(\vec{r},\hat{\Omega}',E') \end{array} \right) \underline{u}(\vec{r}) \\ + \int_{\alpha \in A} \tau_\alpha \delta(\vec{r} - \vec{r}_\alpha^{\,D}) w(E) d\mu_\alpha \end{bmatrix} \\ + q(\vec{r},\hat{\Omega},E) \lambda(\vec{r},\hat{\Omega},E) \end{bmatrix} \tag{8-41}$$

and the augmented design performance functional, i.e., the cost, can also be rewritten equivalently, as:

$$C = \iiint_{\mathcal{D}} dV \left[H(\lambda,\tau_\alpha,\psi,\underline{u},\vec{r}) + \iint_{\Omega} d\Omega \hat{\Omega} \bullet \int_0^{\infty} dE \, \psi \nabla \lambda \right]$$

$$= \underline{c}^T \iiint_{\mathcal{D}} dV \underline{u}(\vec{r}) - \iiint_{\mathcal{D}} dV \int_{\alpha \in A} \tau_\alpha \delta(\vec{r} - \vec{r}_\alpha^{\,D}) D_\alpha^{MAX}(\vec{r}) d\mu_\alpha$$
$$+ \iiint_{\mathcal{D}} dV \iint_{\Omega} d\Omega \int_0^{\infty} dE \begin{pmatrix} \psi(\vec{r},\hat{\Omega},E) \begin{pmatrix} \left(\begin{array}{l} -\underline{\sigma}_t{}^T \lambda(\vec{r},\hat{\Omega},E) \\ + \iint_{\Omega} d\Omega' \int_0^{\infty} dE' \underline{\sigma}_s{}^T(\hat{\Omega} \to \hat{\Omega}', E \to E') \lambda(\vec{r},\hat{\Omega}',E') \end{array} \right) \underline{u}(\vec{r}) \\ + \hat{\Omega} \bullet \nabla \lambda(\vec{r},\hat{\Omega},E) + \int_{\alpha \in A} \tau_\alpha \delta(\vec{r} - \vec{r}_\alpha^{\,D}) w(E) d\mu_\alpha \end{pmatrix} \\ + q(\vec{r},\hat{\Omega},E) \lambda(\vec{r},\hat{\Omega},E) \end{pmatrix} \tag{8-42}$$

8.1.2 The Variation

The variation of the total cost, δC, is next formed as the difference between the costs of two admissible shield designs, either of which may or may not be optimal. In particular, admissibility requires each of the two designs to conform with the doserate limits of 8-29a. Note the argument that differences between the corresponding τ_α multiplier values

or $\lambda(\vec{r},\hat{\Omega},E)$ multiplier functions assigned to the two different designs do not influence the difference in their costs, since each contributes zero to its associated design cost through the terms of Eq.8-32. It is sufficient to consider the direct cost effects of admissible changes in the design, $\underline{u}(\vec{r})$, and the effects of the resulting changes to the angular flux function, $\psi(\vec{r},\hat{\Omega},E)$.

One of the two designs' control function is denoted by the symbol already introduced, i.e., $\underline{u}(\vec{r})$. The other design's control function is denoted as $\underline{u}_o(\vec{r})$. Angular flux functions for the two designs are respectively $\psi(\vec{r},\hat{\Omega},E)$ and $\psi_o(\vec{r},\hat{\Omega},E)$, and their costs are respectively C and C_o. Clearly the second angular flux function must also satisfy the Boltzmann equation in its own right, i.e.,

$$\begin{aligned}
&\hat{\Omega}\bullet\nabla\psi_o(\vec{r},\hat{\Omega},E)+\underline{\sigma}_t{}^T(E)\underline{u}_o(\vec{r})\psi_o(\vec{r},\hat{\Omega},E)\\
&-\oint\!\!\!\!\!\!\int d\Omega'\int_0^\infty dE'\,\underline{\sigma}_s{}^T(\hat{\Omega}'\to\hat{\Omega},E'\to E)\underline{u}_o(\vec{r})\psi_o(\vec{r},\hat{\Omega}',E')-q(\vec{r},\hat{\Omega},E)=0\\
&\text{for}\quad \vec{r}\in\mathcal{D},\ \underline{u}_o\in U;\\
&\text{with vacuum boundary condition:}\quad \psi_o(\vec{r},\hat{\Omega},E)\Big|_{\substack{\vec{r}\in\partial\mathcal{D}\\ \hat{n}\bullet\hat{\Omega}<0}}=0
\end{aligned}\tag{8-43a}$$

and the second design's cost is

$$C_o=\iiint_{\mathcal{D}}dV\left[H(\lambda,\tau_\alpha,\psi_o,\underline{u}_o,\vec{r})+\oint\!\!\!\!\!\!\int d\Omega\,\hat{\Omega}\bullet\int_0^\infty dE\,\psi_o\nabla\lambda\right]$$

$$\begin{aligned}
&=\underline{c}^T\iiint_{\mathcal{D}}dV\,\underline{u}_o(\vec{r})-\int_{\mathcal{D}}dV\int_{\alpha\in A}\tau_\alpha\delta(\vec{r}-\vec{r}_\alpha^{\,D})D_\alpha^{MAX}(\vec{r})d\mu_\alpha\\
&+\iiint_{\mathcal{D}}dV\oint\!\!\!\!\!\!\int d\Omega\int_0^\infty dE\left(\begin{array}{l}\psi_o(\vec{r},\hat{\Omega},E)\left(\begin{array}{l}\left(\begin{array}{l}-\underline{\sigma}_t{}^T\lambda(\vec{r},\hat{\Omega},E)\\ +\oint\!\!\!\!\!\!\int d\Omega'\int_0^\infty dE'\,\underline{\sigma}_s{}^T(\hat{\Omega}\to\hat{\Omega}',E\to E')\lambda(\vec{r},\hat{\Omega}',E')\end{array}\right)\underline{u}_o(\vec{r})\\ +\hat{\Omega}\bullet\nabla\lambda(\vec{r},\hat{\Omega},E)+\int_{\alpha\in A}\tau_\alpha\delta(\vec{r}-\vec{r}_\alpha^{\,D})w(E)d\mu_\alpha\end{array}\right)\\ +q(\vec{r},\hat{\Omega},E)\lambda(\vec{r},\hat{\Omega},E)\end{array}\right)
\end{aligned}\tag{8-43b}$$

The total *variation* between the two designs is an m-vector function of position as follows:

$$\delta\underline{u}(\vec{r})\equiv\underline{u}(\vec{r})-\underline{u}_o(\vec{r}),\tag{8-44a}$$

The variation between their angular flux functions is

$$\delta\psi\left(\vec{r},\hat{\Omega},E\right)\equiv\psi\left(\vec{r},\hat{\Omega},E\right)-\psi_o\left(\vec{r},\hat{\Omega},E\right), \tag{8-44b}$$

and the resulting variation in total shielding cost is:

$$\delta C\equiv C-C_o\equiv$$

$$\iiint_{\mathcal{D}} dV\left[H\left(\lambda,\tau_\alpha,\psi,\underline{u},\vec{r}\right)-H\left(\lambda,\tau_\alpha,\psi_o,\underline{u}_o,\vec{r}\right)+\oiint d\Omega\int_0^\infty dE\,\delta\psi\,\hat{\Omega}\bullet\nabla\lambda\right] \tag{8-44c}$$

Note that, consistent with the earlier comment, Eq. 8-44c does not consider changes in the multipliers between the two admissible designs. This is justified since the multipliers do not contribute directly to the cost.

Since Eq. 8-44c holds true for any variations between admissible shield designs, it in particular holds true for small variations. The limit of infinitesimally small variations, customarily called the *first variation*, can be rewritten using partial derivatives as follows:

$$\lim_{\delta\underline{u}\to\underline{0}}\left[H\left(\lambda,\tau_\alpha,\psi,\underline{u},\vec{r}\right)-H\left(\lambda,\tau_\alpha,\psi_o,\underline{u}_o,\vec{r}\right)\right]=\frac{\partial H}{\partial\underline{u}}\delta\underline{u}+\frac{\partial H}{\partial\psi}\delta\psi \tag{8-45}$$

Then the first variation of the cost performance functional is:

$$\begin{aligned}\delta C&=\iiint_{\mathcal{D}} dV\left[\frac{\partial H}{\partial\underline{u}}\delta\underline{u}+\frac{\partial H}{\partial\psi}\delta\psi+\oiint d\Omega\int_0^\infty dE\left(\hat{\Omega}\bullet\nabla\lambda\right)\delta\psi\right]\\&=\iiint_{\mathcal{D}} dV\left[\frac{\partial H}{\partial\underline{u}}\delta\underline{u}+\left(\frac{\partial H}{\partial\psi}+\oiint d\Omega\int_0^\infty dE\left(\hat{\Omega}\bullet\nabla\lambda\right)\right)\delta\psi\right]\end{aligned} \tag{8-46}$$

or in expanded form,

$$\delta C\equiv\iiint_{\mathcal{D}} dV\left[\begin{array}{l}\left[\begin{array}{l}\underline{c}^T+\\\oiint d\Omega\int_0^\infty dE\left(\begin{array}{l}-\underline{\sigma}_t{}^T\lambda\left(\vec{r},\hat{\Omega},E\right)\\+\oiint d\Omega'\int_0^\infty dE'\,\underline{\sigma}_s{}^T\left(\hat{\Omega}\to\hat{\Omega}',E\to E'\right)\lambda\left(\vec{r},\hat{\Omega}',E'\right)\end{array}\right)\psi\left(\vec{r},\hat{\Omega},E\right)\end{array}\right]\delta\underline{u}\left(\vec{r}\right)\\+\oiint d\Omega\int_0^\infty dE\left[\begin{array}{l}\left(\begin{array}{l}-\underline{\sigma}_t{}^T\lambda\left(\vec{r},\hat{\Omega},E\right)\\+\oiint d\Omega'\int_0^\infty dE'\,\underline{\sigma}_s{}^T\left(\hat{\Omega}\to\hat{\Omega}',E\to E'\right)\lambda\left(\vec{r},\hat{\Omega}',E'\right)\end{array}\right)\underline{u}\left(\vec{r}\right)\\+\hat{\Omega}\bullet\nabla\lambda+\int_{\alpha\in A}\tau_\alpha\delta\left(\vec{r}-\vec{r}_\alpha^{\,D}\right)w(E)d\mu_\alpha\end{array}\right]\delta\psi\left(\vec{r},\hat{\Omega},E\right)\end{array}\right] \tag{8-47}$$

Clearly the first variation of the cost is an approximation to the difference in costs between two admissible shield designs. This approximation is accurate for two admissible designs that are almost identical, but may be less accurate for admissible designs that diverge substantially.

Up until this point the multiplier function, $\lambda\left(\vec{r},\hat{\Omega},E\right)$, has still not been specified except for a smoothness requirement and an external boundary condition. Here a decision is made to choose λ so that the entire coefficient of the $\delta\psi$ term in Eq. 8-47 vanishes identically. This has the practical benefit of eliminating the need to tediously work out relationships between $\delta\underline{u}(\vec{r})$ and $\delta\psi\left(\vec{r},\hat{\Omega},E\right)$. This choice of λ is stated as:

$$\frac{\partial H}{\partial \psi}+\oint\!\!\!\!\!\!\int d\Omega\int_0^\infty dE\left(\hat{\Omega}\bullet\nabla\lambda\right)=0 \tag{8-48a}$$

or in expanded form, as:

$$\begin{aligned}&-\hat{\Omega}\bullet\nabla\lambda\left(\vec{r},\hat{\Omega},E\right)+\underline{\sigma}_t{}^T(E)\underline{u}(\vec{r})\lambda\left(\vec{r},\hat{\Omega},E\right)\\&-\oint\!\!\!\!\!\!\int d\Omega'\int_0^\infty dE'\underline{\sigma}_s{}^T\left(\hat{\Omega}\to\hat{\Omega}',E\to E'\right)\lambda\left(\vec{r},\hat{\Omega}',E'\right)\underline{u}(\vec{r})=\int_{\alpha\in A}\tau_\alpha\delta\left(\vec{r}-\vec{r}_\alpha^{\,D}\right)w(E)d\mu_\alpha\end{aligned} \tag{8-48b}$$

At this point, Eq. 8-48b may be recognized as the familiar defining equation for the adjoint angular flux, with $\lambda\left(\vec{r},\hat{\Omega},E\right)$ serving as the adjoint angular flux variable and with the right hand side of Eq. 8-48b serving as the adjoint flux source terms. Since the adjoint angular flux is commonly denoted by using the same symbol used for angular flux but with a post-superscript symbol affixed, this notation is adopted henceforth, using an asterisk post-superscript, i.e.,

$$\psi^*\left(\vec{r},\hat{\Omega},E\right)\equiv\lambda\left(\vec{r},\hat{\Omega},E\right) \tag{8-49}$$

Then Eq. 8-48b is restated using the preferred adjoint flux notation, as follows:

$$\begin{aligned}&-\hat{\Omega}\bullet\nabla\psi^*\left(\vec{r},\hat{\Omega},E\right)+\underline{\sigma}_t{}^T\underline{u}(\vec{r})\psi^*\left(\vec{r},\hat{\Omega},E\right)\\&-\oint\!\!\!\!\!\!\int d\Omega'\int_0^\infty dE'\underline{\sigma}_s{}^T\left(\hat{\Omega}\to\hat{\Omega}',E\to E'\right)\psi^*\left(\vec{r},\hat{\Omega}',E'\right)\underline{u}(\vec{r})=\int_{\alpha\in A}\tau_\alpha\delta\left(\vec{r}-\vec{r}_\alpha^{\,D}\right)w(E)d\mu_\alpha\end{aligned} \tag{8-50a}$$

The associated boundary condition of Eq.8-37 is also restated:

$$\psi^*\left(\vec{r},\hat{\Omega},E\right)=0,\quad \vec{r}\in\partial\mathcal{D},\hat{n}\bullet\hat{\Omega}>0 \tag{8-50b}$$

When a "detector" within a domain is specified as an adjoint source, the physical significance of the adjoint angular flux is that it states the expected detection effect of introducing one particle per unit time at any particular place, direction, and energy. Note that Eq., 8-50b is the appropriate *vacuum* boundary condition for adjoint angular flux,

since the likelihood of particles ever returning after exiting a convex domain into an infinite external vacuum is indeed zero.

A complicating feature of Eq. 8-50a is that its right hand side does not necessarily represent a single small "detector" location. Instead it represents the entire set of all radiation leakage design constraint locations specified in the shield design problem. That could in principle range from a single constraint for some design problems up to the uncountably infinite number of constraints needed to cover an extended continuous manifold for other design problems. Further complicating the situation, Eq. 8-50a uses weighting multipliers, τ_α ,which have not yet been determined beyond their limitation that ineffective constraint locations where the doserate is less than allowed must have τ_α values of zero. That limitation ensures that ineffective doserate constraints do not act as adjoint angular flux sources. However, the exact nonnegative τ_α values for the effective constraints still remain to be determined.

This development proceeds by next decomposing the adjoint angular flux, $\psi^*\left(\vec{r},\hat{\Omega},E\right)$, into the sum of products of Kuhn-Tucker multipliers with *normalized component* adjoint angular flux functions, $\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right)$, which are hereby defined for this role as follows:

$$\psi^*\left(\vec{r},\hat{\Omega},E\right)\equiv \int_{\alpha\in A} \tau_\alpha \psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right) d\mu_\alpha \tag{8-51}$$

It should be noted that the adjoint Boltzmann equation as stated in Eq. 8-50a is linear, and that its linearity is exploited by the Eq.8-51 decomposition to factor out its measure-space integration and its τ_α multipliers. The physical significance of these Eq.8-51 normalized component adjoint angular flux functions, $\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right)$, is that each states the effect on the doserate at the single location $\vec{r}_\alpha^{\,D}$ of introducing one source particle per unit time at any particular place $\vec{r}$, direction $\hat{\Omega}$, and energy E. This decomposition's practical usefulness is that it allows the normalized component adjoint angular flux functions to be numerically calculated for a shield design before the τ_α Kuhn-Tucker multiplier values are determined. Thus, for any specified shield design, $\underline{u}(\vec{r})$, each normalized component adjoint angular flux function, $\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right)$, is the solution of the following equations, in which all terms are well-defined and known.

$$\begin{aligned}&-\hat{\Omega}\bullet\nabla\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right)+\underline{\sigma}_t{}^T\,\underline{u}(\vec{r})\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right)\\&-\oiint d\Omega'\int_0^\infty dE'\,\underline{\sigma}_s{}^T\left(\hat{\Omega}\to\hat{\Omega}',E\to E'\right)\psi_\alpha{}^*\left(\vec{r},\hat{\Omega}',E'\right)\underline{u}(\vec{r})=\delta\left(\vec{r}-\vec{r}_\alpha^{\,D}\right)w(E)\end{aligned} \tag{8-52a}$$

$$\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right)=0,\quad \vec{r}\in\partial\mathcal{D},\hat{n}\bullet\hat{\Omega}>0 \tag{8-52b}$$

The role of the Kuhn-Tucker multipliers is thus in forming a particular weighted sum over the specified doserate constraint locations $\{\vec{r}_\alpha^{\,D}\}_{\alpha\in A}$ of the effects on their various doserates caused by introducing one particle per unit time at any location $\vec{r}$, direction $\hat{\Omega}$, and energy E.

8.1.3 Optimality

The most important consequence of choosing to set the coefficient of $\delta\psi$ to zero in Eq. 8-47, which led to Eq. .8-48 and its restatement as Eq. 8-50a, is that it simplifies the first variation of the cost function to the following form:

$$\delta C = \iiint_{\mathcal{D}} dV \left[\frac{\partial H}{\partial \underline{u}} \delta \underline{u}(\vec{r}) \right] \tag{8-53}$$

Eq. 8-53 states that with the chosen definition of the Lagrange multiplier as the adjoint angular flux function for designs obeying the doserate constraints, the variation of the shield cost functional between any two slightly different admissible designs can be evaluated by integrating the inner product of the difference $\delta \underline{u}(\vec{r})$ between their design control variables and a sensitivity function, $\frac{\partial H}{\partial \underline{u}}$. This sensitivity function contains the information about radiation attenuation in the shield and about doserate constraints, implicit by its incorporation of the adjoint flux in its definition.

Eq. 8-53 is important because conditions for optimality can be established from it by requiring that no *admissible* change to the control vector function can improve the performance index. A shield design, $\underline{u}(\vec{r})$, is only optimal, i.e., minimizing the cost function *C*, if there are no admissible variations, $\delta \underline{u}(\vec{r})$, resulting in negative values of δC. Deriving necessary conditions from this statement would involve only simple algebra if the Hamiltonian depended nonlinearly on an unconstrained control vector, but in this case its dependence on the control vector is linear and inequality constraints apply. For any spatial regions of an optimal shield design in which the design control variable $\underline{u}(\vec{r})$ is strictly interior to *U* as defined in Eq. 8-16, the admissible variations $\delta \underline{u}(\vec{r})$ could be in any direction in the m-dimensional control space without reaching the boundary of *U*. This implies that all m components of $\frac{\partial H}{\partial \underline{u}}$ must of necessity be zero in such regions. For other regions in which $\underline{u}(\vec{r})$ is on the boundary of *U*, the admissible design control variable variations are restricted by inequalities depending on which of the (m+1) hypersurface boundaries of *U* is constraining. Although the necessary optimality conditions can be elucidated in this way, it involves tedious solution of simultaneous systems of inequalities and it only provides necessary conditions, not sufficient conditions, for optimality.

The preferred alternative approach is to employ Pontryagin's Maximum Principle [Pontryagin et al 1962, 17-22] as proven in his 1962 opus (invoked here as a minimum

principle). The essence of this theorem is that the optimal choice of a control vector *function* defined on an entire (spatial) domain has the property that at each point in the domain it separately optimizes the Hamiltonian over the admissible control set, independent of any considerations for other points in the (spatial) domain. More precisely, Pontryagin's theorem applied to this case asserts that the necessary *and sufficient* condition for optimality of a particular design, $\underline{u}(\vec{r}), \forall \vec{r} \in \mathcal{D}$ is that the optimal design's choice of $\underline{u}$ at each and every spatial point, $\vec{r}$, minimizes the Hamiltonian, H, at that spatial point with respect to all other admissible choices of $\underline{u} \in U$ at that point, with the optimal design's own angular flux and adjoint angular flux functions used to evaluate H.

The Hamiltonian is therefore restated by modifying Eq. 8-41 to use the chosen adjoint angular flux symbol and then rearranging algebraically to isolate the $\underline{u}(\vec{r})$ term:

$$
\begin{aligned}
&H(\psi^*, \tau_\alpha, \psi, \underline{u}, \vec{r}) \equiv \\
&\left[\underline{c}^T + \oiint d\Omega \int_0^\infty dE \psi(\vec{r}, \hat{\Omega}, E) \begin{pmatrix} -\underline{\sigma}_t{}^T(E) \psi^*(\vec{r}, \hat{\Omega}, E) \\ + \oiint d\Omega' \int_0^\infty dE' \underline{\sigma}_s{}^T(\hat{\Omega} \to \hat{\Omega}', E \to E') \psi^*(\vec{r}, \hat{\Omega}', E') \end{pmatrix} \right] \underline{u}(\vec{r}) \\
&+ \oiint d\Omega \int_0^\infty dE q(\vec{r}, \hat{\Omega}, E) \psi^*(\vec{r}, \hat{\Omega}, E) \\
&+ \int_{\alpha \in A} \tau_\alpha \delta(\vec{r} - \vec{r}_\alpha^D) \left[\oiint d\Omega \int_0^\infty dE w(E) \psi(\vec{r}, \hat{\Omega}, E) - D_\alpha^{MAX}(\vec{r}) \right] d\mu_\alpha
\end{aligned}
\tag{8-54}
$$

The Hamiltonian in Eq. 8-54 is written as the sum of an inner product between a position-dependent m-vector and the control m-vector, $\underline{u}(\vec{r})$, plus another term independent of the control vector, i.e., in the following form:

$$H = \underline{b}^T(\vec{r}; \psi^*, \psi) \underline{u}(\vec{r}) + e(\vec{r}, \nu_\alpha; \psi^*) \tag{8-55}$$

Notationally, the semicolon in Eq. 8-55 is introduced here to separate the list of symbols on which a function depends into an initial list for which the dependence is a direct map and a second list on which the dependence is less direct. As stated below, $\underline{b}$ is a mapping from $\vec{r} \in \Re^3$ into $\Re^m$ which also depends on the entire functions $\psi^*(\vec{r}, \hat{\Omega}, E)$ and $\psi(\vec{r}, \hat{\Omega}, E)$:

$$
\begin{aligned}
&\underline{b}(\vec{r}; \psi^*, \psi) \equiv = \\
&\underline{c} + \oiint d\Omega \int_0^\infty dE \left(-\underline{\sigma}_t \psi \psi^* + \psi \oiint d\Omega' \int_0^\infty dE' \underline{\sigma}_s(\hat{\Omega} \to \hat{\Omega}', E \to E') \psi^*(\vec{r}, \hat{\Omega}', E') \right)
\end{aligned}
\tag{8-56}
$$

Similarly,

$$e(\vec{r},\nu_\alpha;\psi^*) \equiv \oiint d\Omega \int_0^\infty dE q(\vec{r},\hat{\Omega},E)\psi^*(\vec{r},\hat{\Omega},E)$$
$$+ \int_{\alpha\in A} \tau_\alpha \delta(\vec{r}-\vec{r}_\alpha^{\,D}) \left[\oiint d\Omega \int_0^\infty dE w(E)\psi(\vec{r},\hat{\Omega},E) - D_\alpha^{MAX}(\vec{r}) \right] d\mu_\alpha \tag{8-57}$$

Eq. 8-56 is not the only way to calculate the $\underline{b}$ vector. In Eq. 8-56 the scattering term is stated as an inner product between forward angular flux and the net out-scattering of adjoint angular flux, i.e., the difference between out-scattering and "total" processes as computed for each material using the adjoint flux. However, it is also equal to the inner product between adjoint flux and the net inscattering of forward flux for the same mix component. This equivalence can be shown formally by changing between iterated integrals and combined higher dimension integration while swapping the dummy variables of integration, as follows:

$$\oiint d\Omega \int_0^\infty dE \left(\psi(\vec{r},\hat{\Omega},E) \oiint d\Omega' \int_0^\infty dE' \underline{\sigma}_s(\hat{\Omega}\to\hat{\Omega}', E\to E')\psi^*(\vec{r},\hat{\Omega}',E') \right) =$$
$$= \oiint d\Omega \int_0^\infty dE \oiint d\Omega' \int_0^\infty dE' \underline{\sigma}_s(\hat{\Omega}\to\hat{\Omega}', E\to E')\psi^*(\vec{r},\hat{\Omega}',E')\psi(\vec{r},\hat{\Omega},E) =$$
$$= \oiint d\Omega' \int_0^\infty dE' \oiint d\Omega \int_0^\infty dE \underline{\sigma}_s(\hat{\Omega}'\to\hat{\Omega}, E'\to E)\psi^*(\vec{r},\hat{\Omega},E)\psi(\vec{r},\hat{\Omega}',E') = \tag{8-58}$$
$$= \oiint d\Omega \int_0^\infty dE \left(\psi^*(\vec{r},\hat{\Omega},E) \oiint d\Omega' \int_0^\infty dE' \underline{\sigma}_s(\hat{\Omega}'\to\hat{\Omega}, E'\to E)\psi(\vec{r},\hat{\Omega}',E') \right)$$

Thus, Eq. 8-57 is equivalently written in terms of either adjoint scattering or forward scattering:

$$\underline{b}(\vec{r};\psi^*,\psi) = \underline{c} +$$
$$\oiint d\Omega \int_0^\infty dE \psi(\vec{r},\hat{\Omega},E) \left(-\underline{\sigma}_t(E)\psi^*(\vec{r},\hat{\Omega},E) + \oiint d\Omega' \int_0^\infty dE' \underline{\sigma}_s(\hat{\Omega}\to\hat{\Omega}', E\to E')\psi^*(\vec{r},\hat{\Omega}',E') \right) \tag{8-59a}$$

or

$$\underline{b}(\vec{r};\psi^*,\psi) = \underline{c} +$$
$$\oiint d\Omega \int_0^\infty dE \psi^*(\vec{r},\hat{\Omega},E) \left(-\underline{\sigma}_t(E)\psi(\vec{r},\hat{\Omega},E) + \oiint d\Omega' \int_0^\infty dE' \underline{\sigma}_s(\hat{\Omega}'\to\hat{\Omega}, E'\to E)\psi(\vec{r},\hat{\Omega}',E') \right) \tag{8-59b}$$

It is useful to substitute the Eq. 8-51 decomposition of the adjoint flux function into Eq. 8-59 and exploit linearity to pass the Kuhn-Tucker adjoint source multipliers through the integral operators. The result is written as follows:

$$\underline{b}(\vec{r};\psi^*,\psi) = \underline{c} + \int_{\alpha\in A} \tau_\alpha \underline{\boldsymbol{d}}_\alpha\left(\vec{r};\psi_\alpha{}^*,\psi\right) d\mu_\alpha \tag{8-60}$$

where $\underline{\boldsymbol{d}}_\alpha\left(\vec{r};\psi_\alpha{}^*,\psi\right)$ is a "discriminant" m-vector function defined for each doserate constraint location $\vec{r}_\alpha^{\,D}$ as follows:

$$\underline{\boldsymbol{d}}_\alpha\left(\vec{r};\psi_\alpha{}^*,\psi\right) \equiv$$
$$\iint d\Omega \int_0^\infty dE\,\psi\left(\vec{r},\hat{\Omega},E\right)\left(-\underline{\sigma}_t(E)\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right) + \iint d\Omega' \int_0^\infty dE'\,\underline{\sigma}_s\left(\hat{\Omega}\to\hat{\Omega}',E\to E'\right)\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right)\right) \tag{8-61a}$$

$$= \iint d\Omega \int_0^\infty dE\,\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right)\left(-\underline{\sigma}_t(E)\psi\left(\vec{r},\hat{\Omega},E\right) + \iint d\Omega' \int_0^\infty dE'\,\underline{\sigma}_s\left(\hat{\Omega}'\to\hat{\Omega},E'\to E\right)\psi\left(\vec{r},\hat{\Omega}',E'\right)\right) \tag{8-61b}$$

The physical significance of this discriminant function can be seen by parsing it. The term, $\left(-\underline{\sigma}_t(E)\psi\left(\vec{r},\hat{\Omega},E\right) + \iint d\Omega' \int_0^\infty dE'\,\underline{\sigma}_s\left(\hat{\Omega}'\to\hat{\Omega},E'\to E\right)\psi\left(\vec{r},\hat{\Omega}',E'\right)\right)$, is an m-vector function of $\left(\vec{r},\hat{\Omega},E\right)$. Its i^{th} component states the net particle source rate at location $\vec{r}$, direction $\hat{\Omega}$, and energy E that would result from inscattering minus removal processes if material number *i* were present at location $\vec{r}$ with volume fraction one. Of course, this interpretation assumes that shielding material is only changed in a very small volume in the immediate neighborhood of location $\vec{r}$ so that the angular flux function ψ remains valid, unchanged by material substitution. This net particle source rate term is then multiplied in the integrand by the normalized component adjoint angular flux function $\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right)$, which itself states the sensitivity of the doserate at location $\vec{r}_\alpha^{\,D}$ to a source particle rate introduced at location $\vec{r}$, direction $\hat{\Omega}$, and energy E. **Therefore, $\underline{\boldsymbol{d}}_\alpha\left(\vec{r};\psi_\alpha{}^*,\psi\right)$, which is defined in Eq. 8-61 as the integral over directions and energy of their product, is the total effect on doserate at location $\vec{r}_\alpha^{\,D}$ of having material number *i* present at location $\vec{r}$ with volume fraction one.**

It is appropriate to remark at this point that this vector discriminant function, $\underline{\boldsymbol{d}}_\alpha\left(\vec{r};\psi_\alpha{}^*,\psi\right)$, can be used to compactly state the doserate constraints on admissible shield design variations. Two nearly identical admissible designs will have identical doserates at each effective radiation constraint location, with each doserate matching the maximum allowable there. It follows that admissible design variations $\delta\underline{u}(\vec{r})$ must obey the following set of integral constraints:

$$\delta D_\alpha \equiv \iiint_{\mathcal{D}} dV \, \underline{\boldsymbol{d}}_\alpha^T\left(\vec{r};\psi_\alpha{}^*,\psi\right)\delta\underline{u}(\vec{r}) = 0 \qquad \forall\,\alpha \in A \tag{8-62}$$

It's also appropriate to remark about how the discriminant affects design optimization. Clearly if shielding material number i is used in a shielding design at location $\vec{r}$ then the i^{th} component of $\underline{\boldsymbol{d}}_\alpha\left(\vec{r};\psi_\alpha{}^*,\psi\right)$ should be negative there. Otherwise a void could be substituted for material number i at that location without compromising radiation attenuation if the i^{th} component of $\underline{\boldsymbol{d}}_\alpha$ were zero there. Substituting a void would actually improve attenuation if the i^{th} component of $\underline{\boldsymbol{d}}_\alpha$ were positive. On the other hand, a negative value of a material's $\underline{\boldsymbol{d}}_\alpha$ at a location does not necessarily mean that it should be used there, since it might be too expensive. Therefore, in Eq.8-60 these $\underline{\boldsymbol{d}}_\alpha$ vector functions are combined with the Kuhn-Tucker multipliers τ_α as weighting factors to which is added the m-vector of material costs per unit volume, $\underline{c}$. The resulting function, $\underline{b}(\vec{r};\psi^*,\psi)$, is multiplied in Eq. 8-55 by the control design vector, $\underline{u}(\vec{r})$, to form the only term in the Hamiltonian that depends directly on $\underline{u}(\vec{r})$.

Pontryagin's theorem, applied in particular to the Hamiltonian function of the present problem as written in Eq. 8-55, asserts that the optimal design control vector $\underline{u}(\vec{r})$ at each point in the spatial domain is the following:

$$\forall\vec{r} \in \mathcal{D}, \qquad \underline{u}_{opt}(\vec{r}) = \underline{\upsilon} \text{ where } \underline{\upsilon} \text{ obeys } \left(\underline{b}^T\left(\vec{r};\psi^*_{opt},\psi_{opt}\right)\underline{\upsilon}\right) = \min_{\underline{\upsilon}\in U}\left(\underline{b}^T\left(\vec{r};\psi^*_{opt},\psi_{opt}\right)\underline{\upsilon}\right) \tag{8-63}$$

This states that if the i^{th} component in the m-vector, $\underline{b}(\vec{r};\psi^*,\psi)$ is negative and more negative than the other components of $\underline{b}$ at location $\vec{r}$ in a shield design, then the i^{th} candidate shield material is the optimal choice for that location. It also states that if no component of $\underline{b}(\vec{r};\psi^*,\psi)$ is positive at a location then the optimal design places a void $(\underline{u} = \underline{0})$ there. If the minimum component of $\underline{b}(\vec{r};\psi^*,\psi)$ is negative then its associated material is optimal, otherwise a void would be better. Thus, simply sorting the computed components of $\underline{b}$ by their numerical values at each location in the domain is sufficient to determine which pure material is optimal there, whether mixtures of pure materials are permitted, or whether a vacuum should he used instead.

Unfortunately, Eq. 8-63 does not explicitly prescribe the optimal mixture if two or more components of $\underline{b}(\vec{r};\psi^*,\psi)$ have equal negative values. If this equality of the two most negative components occurs only at isolated locations then the optimal solution switches there from one pure material to another. However, optimal shields can have regions of positive volume in which the two most negative components of $\underline{b}$ remain equal to each other. That equality between components of $\underline{b}$ requires the Hamiltonian to behave in a

very special way. In such regions the optimal shield design is a mixture of different materials, each with a positive volume fraction less than one, but with the mix fractions varying as a function of position. In control theory the solutions in such regions are termed *singular*, and they can be found analytically by solving other equations derived from these. The control theory derivations proceed by first setting to zero the derivative of the Hamiltonian with respect to the independent variable, which is typically time in most controls problems, and then eliminating any resulting explicit time derivatives by substituting the fundamental or the adjoint differential equations as needed. The analogous analytic approach for singular regions of optimal shield designs would need to contend with all three independent variables of $\vec{r} \in \mathcal{D} \subset \Re^3$, so the derivation of singular equations might start by setting $\underline{0} = \nabla H \equiv \nabla\left[\underline{b}^T(\vec{r};\psi^*,\psi)\underline{u}(\vec{r}) + e(\vec{r},\nu_\alpha;\psi^*)\right]$, then continue by substituting the forward and adjoint Boltzmann equations to eliminate higher order derivatives. The solution $\underline{u}(\vec{r})$ of the resulting complicated partial differential equation, expected to include both $\underline{u}(\vec{r})$ and $\nabla\underline{u}(\vec{r})$ terms, would define how optimal mixtures should vary continuously with position throughout a singular region. But this approach cannot be followed unless the admissible $\underline{u}(\vec{r})$ functions are constrained to have sufficient smoothness so that $\nabla\underline{u}(\vec{r})$ is everywhere well-defined.

The singular region equations are not derived herein, since it is not planned to use them in an optimal shield design computer algorithm in the present work. Experience from the controls field has shown that singular solution algorithms typically require special methods to precisely identify the singular region boundaries, and for some problems there are also special "tangency conditions" which must be satisfied on those boundaries. To avoid this complexity, the approach taken instead relies on the intuitive fact that a uniform mixture of two materials is the limit of a jumbled collection of pure material chunks as the diameter of the largest pure material chunk approaches zero. Indeed, real physical mixtures as opposed to mathematical abstractions are unavoidably chunky, at least on atomic distance scales. It is also obvious that when numerically evaluating Eq. 8-63 at the finite number of predefined discrete locations of a computational grid, the likelihood is vanishingly small of ever calculating two precisely equal negative components of $\underline{b}(\vec{r};\psi^*,\psi)$. By employing only pure materials for $\underline{u}(\vec{r})$ at each predefined discrete location of the shield as dictated by Eq. 8-63, the expected result in singular regions of the solution is a chattering oscillation of $\underline{u}(\vec{r})$ at a high spatial frequency along with a small amplitude oscillation of the Hamiltonian. It is expected that although the resulting chunky and discontinuous $\underline{u}(\vec{r})$ optimal designs are mathematically different from the continuous $\underline{u}(\vec{r})$ optimal designs that might be found by stitching together the smooth solutions of singular and nonsingular regions, that the material costs and attenuation characteristics of the different shield designs would be numerically the same.

It is useful to restate Eq.8-63 in terms of the discriminant function defined in Eq.8-61:

$$\forall \vec{r} \in \mathcal{D}, \qquad \underline{u}_{opt}(\vec{r}) = \underline{\upsilon} \text{ where } \underline{\upsilon} \text{ obeys}$$

$$\left(\left(\underline{c}^T + \int_{\alpha \in A} \tau_\alpha \underline{\boldsymbol{d}}_\alpha^T \left(\vec{r}; \psi_\alpha^*, \psi \right)_{opt} d\mu_\alpha \right) \underline{\upsilon} \right) = \min_{\underline{\upsilon} \in U} \left(\left(\underline{c}^T + \int_{\alpha \in A} \tau_\alpha \underline{\boldsymbol{d}}_\alpha^T \left(\vec{r}; \psi_\alpha^*, \psi \right)_{opt} d\mu_\alpha \right) \underline{\upsilon} \right) \tag{8-64}$$

This completes the development of the Optimal Shield Design Equations.

8.2 Discussion of the Optimal Design Equations

The optimal design equations for the general 3D geometry case whose simultaneous solution yields a minimum-mass optimal radiation shield design are thus Equations 8-28, 8-29, 8-31, 8-52, 8-61, and 8-64. For convenience these are collected together in Figure 8-2. The present section discusses solution strategies, although they have not been tested for this general 3D case. Tested algorithms for the 1D spherical geometry case are described later.

Pontryagin's optimality condition provides a straightforward way to determine whether any particular given radiation shield design is optimal. Substitute that design's $\underline{u}(\vec{r})$ function to specify material volume fractions, solve for the flux functions and the normalized component adjoint flux functions, calculate doserates, and then check whether Pontryagin's optimality condition is everywhere met for some set of Kuhn-Tucker multipliers conforming to their doserate-dependent positivity limitations.

Unfortunately Pontryagin's condition does not provide any algorithm guaranteed to directly *find* the optimum design. The difficulty in applying Eq. 8-64 to find the optimal shield design, $\underline{u}_{\text{opt}}$, is that in order to compute $\underline{\boldsymbol{d}}_\alpha \left(\vec{r}; \psi_\alpha^*, \psi \right)_{opt}$ via Eq. 8-61, one first needs to know both the optimal angular flux function, ψ_{opt} and the optimal component adjoint flux functions, $\left(\psi_\alpha^* \right)_{opt}$. These, in turn, depend on first knowing the optimal shield design, $\underline{u}_{\text{opt}}$ from Eq. 8-63 in order to compute them through Eqs 8-28 and 8-52. Thus, the logical reasoning and the computational progressions are circular.

This type of logical gap has been encountered in many problems for which analytical methods do not provide an algebraic solution. For many such problems, iterative algorithms have been successful. Iterative methods for solving the optimal shielding design equations are therefore sought, since no direct solution strategy presents itself.

In an iterative shield design scheme a sequence of complete shield designs, $\left\{ {}^{[k]}\underline{u}(\vec{r}) \right\}_{k=1}^{\infty}$, is automatically generated, starting from an initial guessed design. The algorithm is successful if the sequence of its generated designs, or a subsequence, converges to a

(1) Forward Boltzmann Equation System:

$$\hat{\Omega} \bullet \nabla \psi\left(\vec{r},\hat{\Omega},E\right)+\underline{\sigma}_t{}^T\left(E\right)\underline{u}\left(\vec{r}\right)\psi\left(\vec{r},\hat{\Omega},E\right)$$

$$-\oiint d\Omega'\int_0^\infty dE'\underline{\sigma}_s{}^T\left(\hat{\Omega}'\to\hat{\Omega},E'\to E\right)\underline{u}\left(\vec{r}\right)\psi\left(\vec{r},\hat{\Omega}',E'\right)-q\left(\vec{r},\hat{\Omega},E\right)=0 \qquad (8\text{-}28)$$

$$\text{for}\quad \vec{r}\in\mathcal{D},\underline{u}\in U;\ \text{with}\ \psi\left(\vec{r},\hat{\Omega},E\right)\Big|_{\substack{\vec{r}\in\partial\mathcal{D}\\ \hat{n}\bullet\hat{\Omega}<0}}=0$$

(2) Doserate Constraints and Evaluations, one for each doserate constraint location:

$$D_\alpha\le D_\alpha^{MAX}\qquad\forall\,\alpha\in A \qquad (8\text{-}29a)$$

$$D_\alpha\equiv\iiint_{\mathcal{D}}dV\oiint d\Omega\int_0^\infty dE\,\delta\left(\vec{r}-\vec{r}_\alpha^{\,D}\right)w\left(E\right)\psi\left(\vec{r},\hat{\Omega},E\right)\equiv\int_0^\infty w\left(E\right)\phi\left(\vec{r}_\alpha^{\,D},E\right)dE \qquad (8\text{-}29b)$$

(3) Normalized Component Adjoint Boltzmann Equation Systems, one for each doserate constraint location:

$$-\hat{\Omega}\bullet\nabla\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right)+\underline{\sigma}_t{}^T\,\underline{u}\left(\vec{r}\right)\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right)$$

$$-\oiint d\Omega'\int_0^\infty dE'\underline{\sigma}_s{}^T\left(\hat{\Omega}\to\hat{\Omega}',E\to E'\right)\psi_\alpha{}^*\left(\vec{r},\hat{\Omega}',E'\right)\underline{u}\left(\vec{r}\right)-\delta\left(\vec{r}-\vec{r}_\alpha^{\,D}\right)w\left(E\right)=0 \qquad (8\text{-}52)$$

$$\text{for}\ \forall\,\alpha\in\text{A},\ \vec{r}\in\mathcal{D},\underline{u}\in U;\ \text{with}\ \psi_\alpha^*\left(\vec{r},\hat{\Omega},E\right)\Big|_{\substack{\vec{r}\in\partial\mathcal{D}\\ \hat{n}\bullet\hat{\Omega}>0}}=0$$

(4) Kuhn-Tucker Multiplier Constraints, one for each doserate constraint location:

$$\tau_\alpha\begin{cases}=0\ \text{if}\ D_\alpha-D_\alpha^{MAX}<0\\ \ge 0\ \text{if}\ D_\alpha-D_\alpha^{MAX}=0\end{cases} \qquad (8\text{-}31)$$

(5) Vector Discriminant Functions, one for each doserate constraint location:

$$\underline{\boldsymbol{d}}_\alpha\left(\vec{r};\psi_\alpha{}^*,\psi\right)\equiv$$

$$\oiint d\Omega\int_0^\infty dE\psi\left(\vec{r},\hat{\Omega},E\right)\left(-\underline{\sigma}_t{}^T\left(E\right)\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right)+\oiint d\Omega'\int_0^\infty dE'\underline{\sigma}_s{}^T\left(\hat{\Omega}\to\hat{\Omega}',E\to E'\right)\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right)\right)$$

$$=\oiint d\Omega\int_0^\infty dE\psi_\alpha{}^*\left(\vec{r},\hat{\Omega},E\right)\left(-\underline{\sigma}_t{}^T\left(E\right)\psi\left(\vec{r},\hat{\Omega},E\right)+\oiint d\Omega'\int_0^\infty dE'\underline{\sigma}_s{}^T\left(\hat{\Omega}'\to\hat{\Omega},E'\to E\right)\psi\left(\vec{r},\hat{\Omega}',E'\right)\right) \qquad (8\text{-}61)$$

(6) Pontryagin's Optimality Condition:

$$\forall\vec{r}\in\mathcal{D},\qquad \underline{u}_{opt}\left(\vec{r}\right)=\underline{\upsilon}\ \text{ where}\ \underline{\upsilon}\ \text{ obeys}$$

$$\left(\left(\underline{c}^T+\int_{\alpha\in A}\tau_\alpha\underline{\boldsymbol{d}}_\alpha^T\left(\vec{r};\psi_\alpha{}^*,\psi\right)\Big|_{opt}d\mu_\alpha\right)\underline{\upsilon}\right)=\min_{\underline{\upsilon}\in U}\left(\left(\underline{c}^T+\int_{\alpha\in A}\tau_\alpha\underline{\boldsymbol{d}}_\alpha^T\left(\vec{r};\psi_\alpha{}^*,\psi\right)\Big|_{opt}d\mu_\alpha\right)\underline{\upsilon}\right) \qquad (8\text{-}64)$$

Figure 8-2: Summary of the Optimal Shield Design Equations

minimum-cost optimal shield design, i.e., ${}^{[k]}\underline{u}(\vec{r}) \rightarrow {}^{[\infty]}\underline{u}(\vec{r}) = \underline{u}_{opt}(\vec{r})$, consistent with the specified radiation leakage doserate constraints. Iterative methods must also include other features in addition to their sequential loop algorithms. They need to allow for initial guesses, either synthesizing them automatically or accepting an operator input. They need to recognize convergence within a finite number of steps and halt loop execution when the result is sufficiently close to the solution. They need to recognize a failure to converge and halt loop execution in that circumstance with appropriate flags set. However, the present discussion focuses on the iteration scheme.

Iterative methods can be generally summarized by the problem of numerically solving the equation $f(x) = y$. Arbitrarily choose another function, $g(\bullet)$, then define a third function in terms of these two, as $h(x) = x + g(y) - g(f(x))$. Then any x satisfying $f(x) = y$ obviously satisfies $x = h(x)$, and if $g(\bullet)$ is invertible then the converse is also true. Iterative algorithms simply replace the equality, $x = h(x)$, with the sequential algorithm, ${}^{[k+1]}x = h({}^{[k]}x)$, where *k* is the sequential term number. Starting from an initial guess the infinite sequence which this generates may or may not converge, but if it does, i.e., if ${}^{[k]}x \rightarrow {}^{[\infty]}x$, then it necessarily converges to a solution of $f(x) = y$, i.e., to $f({}^{[\infty]}x) = y$ The *art* of iterative methods is in choosing $g(\bullet)$ to promote convergence.

The Newton-Raphson method is a well-known example of an iteration scheme. Here, x represents a real number and $g(\bullet)$ is chosen as division by the derivative of $f(x)$. For many functions this method provides extremely fast convergence for close initial guesses but it may altogether fail to converge for less accurate initial guesses. Simpler schemes not requiring evaluation of derivatives may converge more slowly but for a wider range of initial guesses. Newton's method can be extended to situations where x represents an n-component vector of real numbers by choosing $g(\bullet)$ as multiplication by the inverse of the Jacobian matrix of partial derivatives of $\underline{f}(\underline{x})$. However, this is not usually attempted for very large n since the cost of evaluating n^2 partial derivatives and inverting an n-by-n matrix during each iteration becomes prohibitive. Simpler iterative schemes are typically used if n is large.

In the present problem the "x" solution that is sought is the complete optimal shield design, $\underline{u}_{opt}(\vec{r})$, which is defined everywhere throughout the continuous 3D problem domain. In principle, an infinite number of real values is needed to specify $\underline{u}_{opt}(\vec{r})$. For practical applications in which $\underline{u}_{opt}(\vec{r})$ may be approximated by discrete sampled values, the finite number of real values needed to approximate it may be large. Therefore, it seems wise to only consider the simplest possible iteration schemes.

Perhaps the simplest way to produce a redesign iteration scheme is to break apart Pontryagin's optimality condition between successive designs. Instead of the optimality

condition being self-referential, the broken-apart version specifies the next sequential design in terms of its predecessor design. The sequential algorithm replacing the Eq. 8-64 equality is as follows:

$$\forall \vec{r} \in \mathcal{D}, \quad {}^{[k+1]}\underline{u}(\vec{r}) = \underline{\upsilon} \text{ where } \underline{\upsilon} \text{ obeys}$$
$$\left(\left(\underline{c}^T + \int_{\alpha \in A} \tau_\alpha \underline{\boldsymbol{d}}_\alpha^T \left(\vec{r}; {}^{[k]}\psi_\alpha{}^*, {}^{[k]}\psi \right) d\mu_\alpha \right) \underline{\upsilon} \right) = \min_{\underline{\upsilon} \in U} \left(\left(\underline{c}^T + \int_{\alpha \in A} \tau_\alpha \underline{\boldsymbol{d}}_\alpha^T \left(\vec{r}; {}^{[k]}\psi_\alpha{}^*, {}^{[k]}\psi \right) d\mu_\alpha \right) \underline{\upsilon} \right) \tag{8-65}$$

Here ${}^{[k]}\psi$ and ${}^{[k]}\psi_\alpha{}^*$ are respectively calculated from Eqs 8-28 and 8-52 using ${}^{[k]}\underline{u}(\vec{r})$ as their assumed design. Eq. 8-65 clearly eliminates the circular logic of the optimality condition, replacing it with a computable sequential algorithm. It is henceforth adopted for redesign iteration schemes considered herein, along with the obvious uses of Eqs 8-28 and 8-52. However, two iteration scheme issues remain concerning the Kuhn-Tucker parameters, i.e., their number and the sharpness of their effects.

In all but pathological cases, it is impossible for finite numerical calculations using a digital computer to solve an infinite number of equations. Although the $\alpha \in A$ indexing notation introduced in Eqs 8-29 can abstractly accommodate uncountably infinite numbers of precise radiation constraints, actual computer calculations are limited to a finite number of them. This point seems consistent with the observation that the usefulness of mathematical results involving infinities is often theoretical. Arbitrary radiation dose constraints considered in a computer program can be accommodated for a finite number of discrete locations. They could be modeled over the infinite number of points of a continuum by adopting a different representation using, e.g., finite elements or a truncated harmonic series, or perhaps wavelets. However, in each such representation the number of values used in a computer is finite, although not bounded theoretically. Thus, prior to initiating any iterative calculation scheme for optimal shield design, a *finite* number *N* of doserate constraint locations should be selected for the problem. One could in advance eliminate all locations whose doserate constraints are clearly ineffective based on other constraints being effective in retained locations. Other strategies could exploit symmetry. Thus, for the present discussion, Kuhn-Tucker adjoint source multipliers, τ_α, are assumed to be only a finite number of nonnegative real values, and the abstract measure space integration over doserate constraint locations, $\int_{\alpha \in A} (\bullet) d\mu_\alpha$, can be replaced by the finite sum, $\sum_{\alpha \in A} (\bullet)$.

On the issue of the sharpness of Kuhn-Tucker multiplier effects, it is important to recognize that the adjoint source multipliers, τ_α, are extremely constrained by very exact limits on doserate. If the doserate at any prescribed radiation constraint point, $\vec{r}_{\alpha 0}^{\,D}$ is slightly less than the allowable doserate there, $D_{\alpha 0}^{MAX}$, even $\varepsilon = 10^{-100}$ less, then Eq. 8-31

requires $\tau_{\alpha 0}$=0. The only permitted way for a particular $\tau_{\alpha 0}$ value to be positive is for the radiation dose at that location to *precisely* match the allowable dose there. On the other hand, if the radiation dose at $\vec{r}_{\alpha 0}$ is even slightly greater than $D_{\alpha 0}^{MAX}$, even $\varepsilon=10^{-100}$ more, then that entire solution must be discarded as inadmissible since it violates the inequality in Eq. 8-29a. Such precision is not compatible with computer calculations using floating point numbers. Worse, if Eqs 8-31 and 8-65 were implemented directly then successive designs might oscillate instead of converging. Thus, a different approach is taken in which the algorithm "backs into" the proper multiplier values.

Consider the proposed "Inner Loop" iterative shield redesign algorithm depicted in Figure 8-3. It includes in a single loop the solution of Eq. 8-28, the evaluation of Eq. 8-29a, the solution of Eq. 8-52, the evaluation of Eq. 8-61, and the evaluation of Eq. 8-65 before incrementing the redesign sequence number, k. All Kuhn-Tucker multiplier values, τ_α, are set outside this loop to constant unchanging values. The resulting sequence of shield designs calculated by this loop has an associated sequence of cost values and, for each specified constraint location, an associated sequence of radiation doserates. Because of the lack of uniqueness for discontinuous designs, "convergence of the design to a limiting design", $^{[k]}\underline{u}(\vec{r}) \rightarrow {}^{[\infty]}\underline{u}(\vec{r})$, is interpreted here to mean simply that these real number sequences for costs and doserates converge to their limit values. Assuming the inner loop converges in this sense, then the sequence of doserates, $\left\{{}^{[k]}D_\alpha\right\}_{k=1}^{\infty}$ at each constraint location must converge to some *particular* limiting doserate value, ${}^{[\infty]}D_\alpha$.

The Figure 8-3 flowchart of an iterative inner loop is schematic only, e.g., it omits any details of algorithms detecting either convergence or its failure. However, it does define algorithms for generating the next shield design from its predecessor design, thus allowing an infinite sequence of shield designs to be automatically generated. Its algorithms include solving the Forward and Adjoint Boltzmann equations, calculating doserates at the radiation constraint locations, calculating vector discriminant functions, and implementing the sequential version of Pontryagin's optimality condition, Eq. 8-65. However, the Kuhn-Tucker parameters are not adjusted within this loop and the doserate constraints are ignored.

Kuhn-Tucker parameters τ_α are used within the loop as weighting factors for the discriminant vector functions of position that determine the next sequential design. Thus, the chosen τ_α and the chosen initial design together determine the entire infinite sequence of designs generated. It seems to be a plausible conjecture that the τ_α values determine the limiting design to which the sequence might converge, and that along with it they determine the limiting doserates. It is easy to show that any limiting design to which the Figure 8-1 loop converges would be optimal for a slightly different shield design problem than the one specified. The difference would be that the allowable maximum doserate values, D_α^{MAX}, would instead need to have been specified equal to the limiting doserates

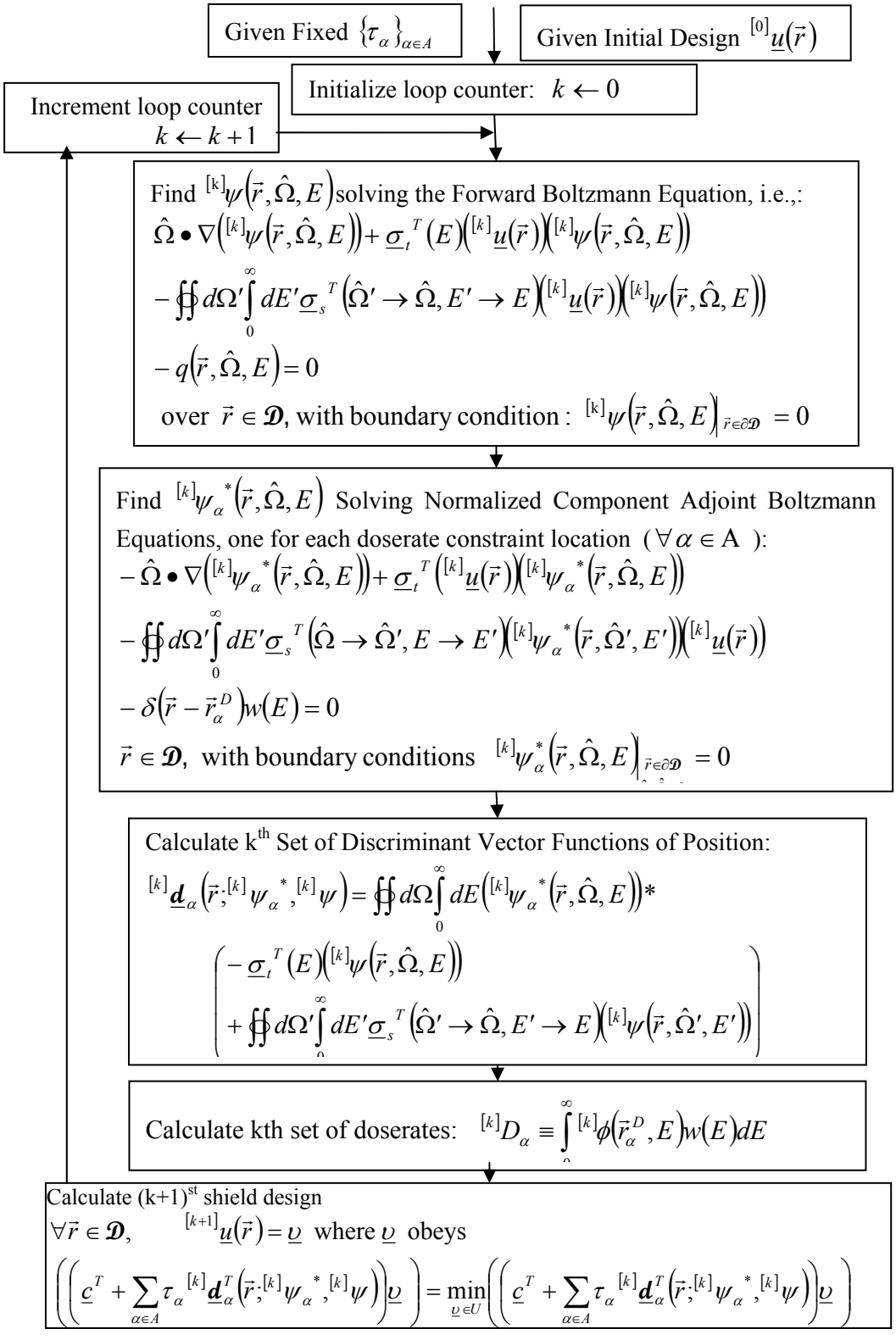


Figure 8-3: Shield Redesign Inner Loop Basic Iteration Scheme

actually calculated by the inner loop's iterations, ${}^{[\infty]}D_\alpha$. The limiting design would then automatically meet these different doserate constraints as required by Eq.8-29a. The nonnegative Kuhn-Tucker multipliers chosen outside the loop would then automatically be in conformance with the Eq. 8-31 positivity restrictions. The limiting forward angular flux function and the limiting normalized component adjoint angular flux functions would then automatically satisfy their forward and adjoint Boltzmann equations, respectively Eqs 8-28 and 8-52. To the extent that the $\underline{u}(\vec{r})$ design function converges, ${}^{[k]}\underline{u}(\vec{r}) \to {}^{[\infty]}\underline{u}(\vec{r})$, Pontryagin's optimality condition Eq. 8-64 would automatically be satisfied for the limiting design.

Thus, if it converges, the proposed Fig. (8-1) loop algorithm would map from an *N*-vector of assumed Kuhn-Tucker multiplier τ_α values to an *N*-vector of limiting doserates ${}^{[\infty]}D_\alpha$, along with an accompanying minimum-cost optimal shield design consistent with the allowable maximum doserates, D_α^{MAX} having been set equal to ${}^{[\infty]}D_\alpha$ instead of their actually specified values. Thus, unlimited iterations to convergence of the entire Figure 8-1 loop can be considered equivalent to evaluating one N-vector function of an N-vector argument, e.g., $\underline{f}(\underline{x})$.

What is needed to finish solving the original shield design optimization problem is a numerical way to find a solution $\underline{x}$ satisfying the N-vector equation $\underline{f}(\underline{x}) = \underline{y}$, where $\underline{x}$ represents the τ_α values and $\underline{y}$ represents the actually specified doserates. Such methods exist and are available in numerical software as *numerical equation solving* algorithms. A variety of methods are used in such solvers, e.g., Newton's method using approximate Jacobian matrices estimated from finite differences. Each solver operates iteratively, passing $\underline{x}$ vectors to a different software module implementing the $\underline{f}(\underline{x})$ function evaluations. Each records the historical sequence of evaluation results in order to build a rough map of $\underline{f}(\underline{x})$ sufficient to adjust results towards the specified $\underline{y}$.

Thus, a complete algorithm to solve the shield design optimization problem with maximum doserate limits as specified can be configured by embedding the entire Figure 8-3 inner loop within a "numerical equation solving" outer iterative loop, as illustrated in Figure 8-4. The outer loop sets the *N*-vector of τ_α values, observes the resulting *N*-vector of limiting ${}^{[\infty]}D_\alpha$ values returned after convergence of the inner loop, and sequentially readjusts the τ_α values in such a way that the resulting limiting ${}^{[\infty]}D_\alpha$ eventually approach the D_α^{MAX} doserate limits actually specified.

8.2.1 Convergence Arguments

Although numerical equation solvers available to use in the proposed Figure 8-2 shield redesign outer loop algorithm have user experience bases attesting to their reliability in

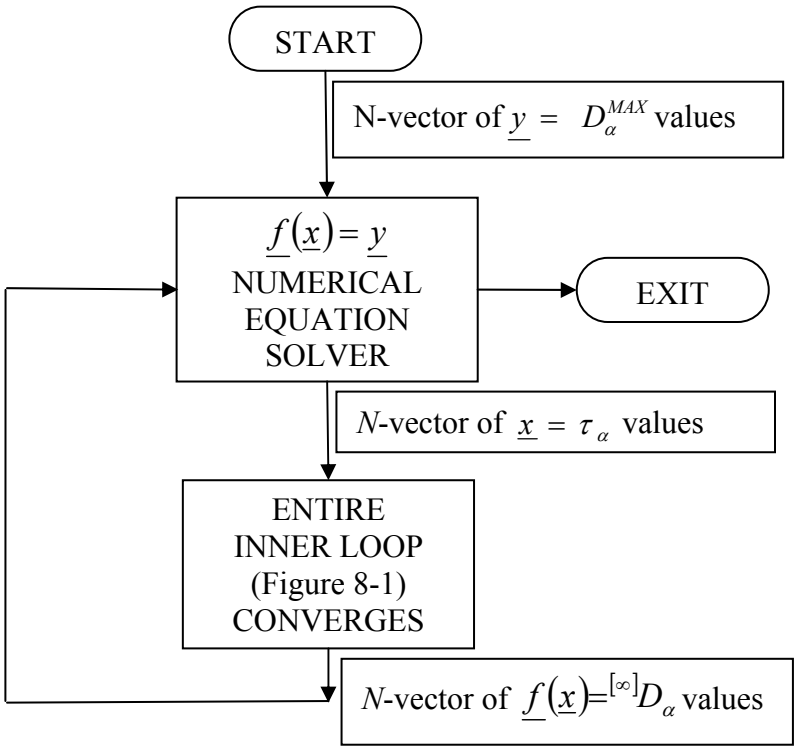


Figure 8-4: Shield Redesign Outer Loop Basic Iteration Schemee

converging to solutions, the overall scheme relies also on convergence of the proposed Figure 8-1 inner iteration loop. It has no similar experience base. It therefore seems prudent to investigate the conditions under which the inner loop is likely to converge.

An infinite sequence of real numbers is said to be monotonic if either every successor term is not greater than its predecessor term or if every successor term is not less than its predecessor term. An elementary theorem states that every bounded monotonic sequence is convergent. Although the converse does not hold true, i.e., some nonmonotonic sequences are also convergent, this theorem may provide an argument relevant to the inner loop iteration.

The following arguments concern a sequence of real numbers, each one of which is calculated as the volume integral of the Hamiltonian for a shield design in the automatically generated sequence of shield designs, i.e., as $\left\{ \iiint_{\mathcal{D}} H\left({}^{[k]}\psi^*, {}^{[k]}\psi, {}^{[k]}\underline{u}, \vec{r}\right) dV \right\}_{k=1}^{\infty}$, where the adjoint flux is taken as the weighted sum of normalized component adjoint flux functions using the fixed Kuhm-Tucker parameter values, τ_α. Note that the difference between successive values of this sequence is not equal to δC, since Pontryagin's rule as implemented in the Figure 8-1 algorithm does not

conform with Eq. 8-62. Instead, the algorithm allows doserates to change, finding their limiting values as determined by the τ_α parameters.

Local changes in the Hamiltonian due to shield redesign are proportional to the difference between material unit costs plus a Kuhn-Tucker parameter weighted sum of differences in projected doserates. The shield redesign algorithm Eq. 8-65 chooses for each location in the shield whichever material minimizes the Hamiltonian there, with the calculation using the angular flux and adjoint angular flux calculated for the earlier of the two shield designs. Thus, at each location the redesign algorithm only changes the shield material if that change would reduce the Hamiltonian below the previous design's Hamiltonian value there. This guarantees that the difference between the Hamiltonians of two successive designs, when calculated using the earlier design's forward and adjoint flux functions, is everywhere nonpositive. It follows that the difference between volume integrals of the Hamiltonians is also nonpositive, at least when both are calculated using the earlier design's forward and adjoint angular flux functions.

If no successor design in the sequence could ever have a higher Hamiltonian volume integral than its predecessor, then the sequence of real numbers would by definition be monotone decreasing. For any particular set of Kuhn-Tucker parameter values a finite lower bound can easily be identified which the sequence cannot possibly reach. This points towards a conclusion that the sequence would necessarily be convergent. Although convergence of this integral does not guarantee through a simple argument that the shield design and its doserates would converge with it, it is difficult to imagine how they could avoid converging since they are not independent.

8.2.2 Possible Algorithm Modifications To Promote Convergence

However, there is a logical gap remaining, since the redesign's change in the Hamiltonian as stated above is only a linear approximation using the first derivative. The full difference between Hamiltonian volume integrals of successive shield designs also includes higher order terms not shown in Eq. 8-47. If the Hamiltonian integrals were evaluated twice within each inner loop iteration, i.e., both before and also after the loop's shield redesign, then these higher order terms would appear as a difference between the second evaluation of one loop and the first evaluation of the next. Although both of these integral evaluations would be for the same shield design, one of them would use the various flux functions calculated for its predecessor design. The logical gap is that if these higher order terms are too large and of the wrong sign then they might cause the sequence of values to alternate, compromising sequence monotonicity and thus removing the guarantee of convergence.

The algebraic simplicity of the Hamiltonian's form in, e.g., Eq. 8-54, allows the higher order terms (H.O.T.) neglected in the first variation to be precisely stated as follows:

$$H.O.T. =$$

$$\iiint_{\mathcal{D}} dV \oiint d\Omega \int_0^\infty dE \left(\begin{array}{l} -\underline{\sigma}_t{}^T(E)\psi^*\left(\vec{r},\hat{\Omega},E\right) \\ +\oiint d\Omega' \int_0^\infty dE' \underline{\sigma}_s{}^T\left(\hat{\Omega}\to\hat{\Omega}', E\to E'\right)\psi^*\left(\vec{r},\hat{\Omega}',E'\right) \end{array} \right) \left(\delta\underline{u}(\vec{r})\right)\left(\delta\psi\left(\vec{r},\hat{\Omega},E\right)\right)$$

This iterated integral involves the product of two difference functions, i.e., the change in the design control function and the change in the angular flux function. Since for two nearly identical designs each of these is small, it follows that the integral of their product would then be negligible. This suggests a simple strategy for ensuring convergence of the Figure 8-1 shield redesign inner loop algorithm. Make sequential changes in the design small, since if they are kept sufficiently small then convergence is guaranteed.

A difficulty with this strategy is that the sequential version of Pontryagin's principle restated as a shield redesign rule in Eq. .8-65 does not restrict the extent of the changes between one shield design and its successor. It does not even consider the preceding shield design explicitly, whose details therefore only affect its successor design through effects on the calculated forward and adjoint angular flux functions. Using this Eq. 8-65 algorithm without modification, the only way to keep sequential design changes small would be for the user to supply initial designs that are almost identical to the optimal design. That restriction would not be useful.

However, the Eq. 8-65 algorithm derived from Pontryagin's Principle can be trivially modified by replacing its initial clause, $\forall\vec{r}\in\mathcal{D}$. Instead of each iteration replacing the shielding material everywhere that a better material choice is indicated, restrict improvements to a proper subset of the computational domain. In the rest of the domain, retain the previous design's material choices without change. This modified algorithm could be stated mathematically as follows:

$$\begin{array}{l} \forall\vec{r}\in\mathcal{D},\quad \underline{v}(\vec{r}) = \underline{\upsilon} \text{ where } \underline{\upsilon} \text{ obeys} \\ \left(\left(\underline{c}^T + \sum_{\alpha\in A}\tau_\alpha{}^{[k]}\underline{\boldsymbol{d}}_\alpha^T\left(\vec{r};{}^{[k]}\psi_\alpha{}^*,{}^{[k]}\psi\right)\right)\underline{\upsilon}\right) = \min_{\underline{\upsilon}\in U}\left(\left(\underline{c}^T + \sum_{\alpha\in A}\tau_\alpha{}^{[k]}\underline{\boldsymbol{d}}_\alpha^T\left(\vec{r};{}^{[k]}\psi_\alpha{}^*,{}^{[k]}\psi\right)\right)\underline{\upsilon}\right) \\ \textit{Then} \\ \forall\vec{r}\in\mathcal{E}_k\subset\mathcal{D},\quad {}^{[k+1]}\underline{u}(\vec{r}) = \underline{v}(\vec{r}) \\ \forall\vec{r}\in\mathcal{D}\text{ - }\mathcal{E}_k,\quad {}^{[k+1]}\underline{u}(\vec{r}) = {}^{[k]}\underline{u}(\vec{r}) \end{array} \qquad (8\text{-}66)$$

This algorithm can be implemented for any arbitrary subset of the spatial domain, $\mathcal{E}_k\subset\mathcal{D}$. The subscript, k, is used here to emphasize the fact that the subset of the spatial domain in which redesign changes are allowed could also be changed from iteration to iteration. If $\mathcal{E}_k$ is chosen as a small enough subset of the domain, the resulting changes between two successive shield designs can be kept arbitrarily small. Regardless of how the subset $\mathcal{E}_k\subset\mathcal{D}$ is chosen, the modified Eq. 8-66 algorithm inherits the property from

Eq.8-65 that the volume-integrated Hamiltonians of successive designs, calculated using the various flux functions of the earlier design, cannot possibly increase.

One appealing strategy would be to redesign only the part of the shield that deviates from optimality in the most *egregious* way. A measure of the local deviation from optimality is the size of the reduction in the Hamiltonian that would accompany redesign if allowed, i.e., $\left(\underline{c}^T + \sum_{\alpha\in A} \tau_\alpha{}^{[k]}\underline{\boldsymbol{d}}_\alpha^T\left(\vec{r};{}^{[k]}\psi_\alpha{}^*,{}^{[k]}\psi\right)\right)\left({}^{[k]}\underline{u}(\vec{r})-\underline{v}(\vec{r})\right)$. This strategy can be expressed in mathematical form by defining the subset in which redesign is permitted as follows:

$$\boldsymbol{\mathcal{E}}_k = \left\{\vec{r}\in\boldsymbol{\mathcal{D}}:\quad \left(\underline{c}^T + \sum_{\alpha\in A} \tau_\alpha{}^{[k]}\underline{\boldsymbol{d}}_\alpha^T\left(\vec{r};{}^{[k]}\psi_\alpha{}^*,{}^{[k]}\psi\right)\right)\left({}^{[k]}\underline{u}(\vec{r})-\underline{v}(\vec{r})\right)\geq \varepsilon_k\right\} \tag{8-67}$$

Here, ε_k would be a nonnegative number selected by the user to serve as a threshold for deciding whether departures from optimality are severe enough to merit changing them during the current redesign iteration. Setting ε_k to zero would permit all of the shield to be redesigned in the current iteration, making Eq. 8-66 match the performance of Eq. 8-65. A large value of ε_k would prevent any redesign changes at all, an unwanted outcome. An equivalent way to achieve this strategy without any chance of setting ε_k too large would be to sort the calculated values of

$$\left(\underline{c}^T + \sum_{\alpha\in A} \tau_\alpha{}^{[k]}\underline{\boldsymbol{d}}_\alpha^T\left(\vec{r};{}^{[k]}\psi_\alpha{}^*,{}^{[k]}\psi\right)\right)\left({}^{[k]}\underline{u}(\vec{r})-\underline{v}(\vec{r})\right) \tag{8-67a}$$

for the entire domain into ascending order, and then choose the most egregious user-selected *fraction* of them for redesign.

An entirely different strategy is to modify $\underline{u}(\vec{r})$ at each location by a limited m-space "distance" step to a new $\underline{u}(\vec{r})$ minimizing the Hamiltonian there consistent with the nonnegativity constraints on volume fractions. Although conceptually easy to implement, this alternative strategy would require the discrete ordinates code be able to accommodate arbitrary mixtures of component materials during its successive iterations.

General Statement of 1D Spherical Geometry Optimal Shield Design Problem

It is a great simplification both conceptually and practically to reduce dimensionality from three dimensions to only one. The 1D spherical shell radiation shield design problem is to choose a set of concentric spherical shell layers such that together they provide a required reduction of the external radiation dose resulting from a central radiation source. Spherical shield designs specify the inner and outer radii of each of their shell layers and choose each layer's composition from a set of admissible shield materials. Each design which satisfies the radiation attenuation requirement has its own total material cost, calculated as the sum over its shell layers of the products of each shell layer volume times its mass density times the cost per unit mass assigned to its material type. Within the set of designs meeting the attenuation requirement, a shield design is optimal if there are no others having less total material cost.

8.3 Spherically Symmetric Minimum-Mass Optimal Shield Design Algorithms

Spherically symmetric design optimization problems, schematically represents by Figure 8-5, are a special subset of general 3D problems. Like the general 3D problems their problem statements are in the Table 8-1 form, their solutions must satisfy the Table 8-2 conditions, and the algorithms of Figures 8-3 and 8-2 could be used to iteratively find their solutions. However, their situation is simpler than the general 3D case. A 1D spherical shield design is easier to describe since a spherically symmetric shield design, $\underline{u}(r)$, requires much less information to specify than $\underline{u}(\vec{r})$ for a general 3D shield design. Positions in a sphere are described by radial distances rather than vectors. Since the radiation doserate constraint is specified on a spherical surface enclosing the shield and since all locations on that surface receive identical radiation doses, it follows that there is only one independent doserate constraint. The $\alpha \in A$ index notation used in the 3D case to denote different radiation doserate constraint locations can therefore be omitted altogether. There is only a single Kuhn-Tucker multiplier , τ. This implies in turn that for nontrivial design problems the doserate constraint will be met as an equality, i.e., the constraint will certainly be effective, so in the solution, τ cannot be forced to be zero as ineffective Kuhn-Tucker parameters are in 3D problems. The adjoint source distribution, for the normalized component adjoint angular flux which τ multiplies, is shaped as an enclosing spherical surface. And since there is only one Kuhn-Tucker parameter to determine, t, the proposed outer loop in Figure 8-4 becomes a 1D search algorithm.

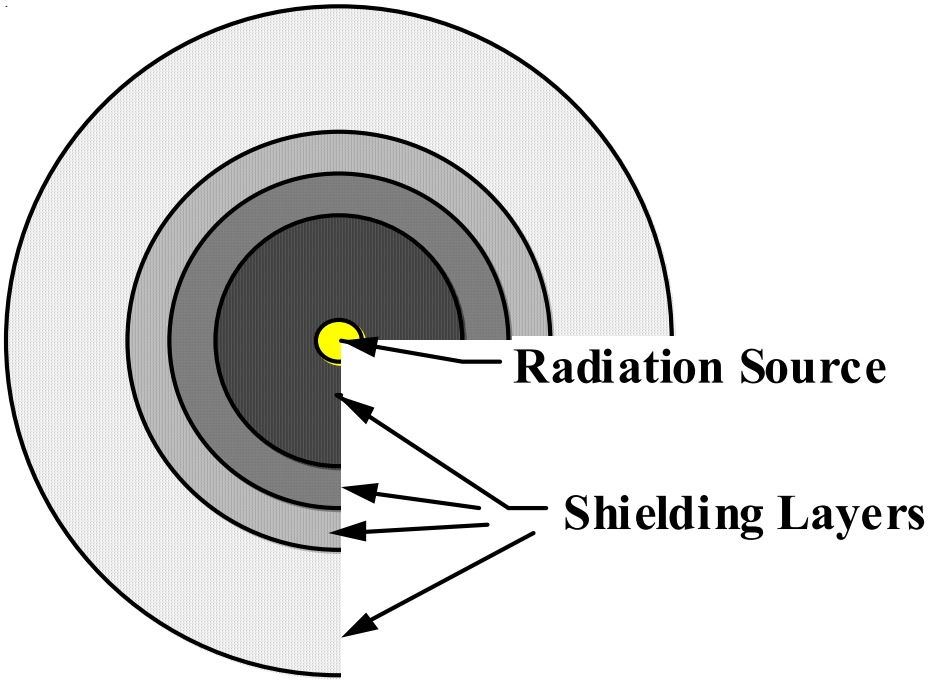


Figure 8-5: Spherical Shielding Configuration

8.3.1 Use of the SCALE System

It was decided that for the present effort, a new software module to automatically design minimum cost (or mass) 1D spherical optimal radiation shields should be coded as an addition to the SCALE (**S**tandardized **C**omputer **A**nalyses for **L**icensing **E**valuation) software system. As stated in the SCALE Version 5 manual, "Since the initial release of SCALE in 1980, the code system has been widely used for evaluation of nuclear fuel facility and package designs." By now, the SCALE system has been extensively benchmarked and has become well established with an extensive group of users. The SCALE system includes a variety of "functional module" codes performing fission criticality, transmutation, radiation propagation, and shielding calculations. SCALE has multiple master cross section libraries, including coupled neutron/gamma libraries. It has material properties libraries along with code modules to prepare material information in formats suitable for use by other modules. It has a library of utility programs. It has a library of special FORTRAN subroutines available for use by a new software module.

SCALE includes a set of "control module" codes, each of which causes subsets of SCALE's functional modules to execute in predefined sequences while also accomplishing associated auxiliary calculations. Some control modules such as e.g., the SAS1 module, implement several selectable sequences, one of which is the SAS1X sequence. SCALE users select any particular sequence by typing its name at the start of their prepared input data file. One of the existing SCALE control sequences, SAS1X, was found to be similar to the sequence of calculations needed to design optimal radiation shields. The SAS1X module's code was therefore adopted as a starting point. By adding modifications, it was adapted to develop the new SAS1XOPT control module which implements the SAS1XOPT control sequence.

It was decided for simplicity's sake to combine the two loops proposed in Figures 8-1 and 8-2 into a single loop for this 1-D case. The single loop approach appeared to be best for this provisional experimental exploratory code since it would avoid the need to design and test reliable algorithms to recognize convergence or its failure in the Figure 8-1 inner loop before returning to the Figure 8-2 outer loop, allowing the resulting single loop's executions to instead be terminated by a simple iteration loop count limit. As shown in Figure 8-1, the job of the inner loop is to make improvements in the cost effectiveness of material assignments within the shield by replacing one assigned material by another in regions where the Eq. 8-65 rule derived from Pontryagin's principle so dictates. As shown in Figure 8-2, the job of the outer loop is to adjust Kuhn-Tucker parameters controlling shield size in order to bring the resulting external doserates in conformance with their specified maximum allowed limits. The loops' combination into a single loop raised the concern that these two jobs might interfere with each other. The approach taken to separate them was to perform them at different speeds. The single loop's implementation of Eq. 8-65 can replace the entire shield design in a single iteration (although the slower Eq. 8-66 algorithm is available if needed to promote convergence). The single loop's action driving the external doserate to match its specified value occurs over the course of many loop iterations.

The single loop's algorithm to control external doserate is quite simple. After each shield redesign, each iteration and the evaluation of its radiation attenuation performance by running the discrete ordinates code, XSDRNPM, the Kuhn-Tucker parameter, τ, is multiplied by a factor that reflects the mismatch between the resulting leakage doserate and the required doserate specified by the user. The algorithm is as follows:

$$\tau^{(k+1)} = \tau^{(k)} \left(\frac{doserate^{(k)}}{doserate\_reqd} \right)^{\gamma}$$

It can be shown that in terms of log-ratios of doserates, this algorithm functions similarly to integral feedback as used in servo loop implementations. The loop can be made to function slowly by choosing a small value of γ (which is actually implemented in the new code as the user-input parameter, p01). Integral feedback has the property that it continues changing until it reaches an error-nulling condition, thus making it useful for controlling nonlinear processes whose nonlinear relations are not well characterized. That is indeed the situation here, where the nonlinear relation between the Kuhn-Tucker parameter value and the resulting shield's leakage doserate is uncertain.

Figure 8-6 shows the sequence of calculations performed in the new SAS1XOPT sequence. The blocks shown in color were already part of the SAS1X sequence, while the remaining blocks represent new code.

8.3.2 Some SAS1XOPT Iterative Algorithm Details

(0) After SCALE's Material Information Processor reads the list of admissible material mixes from the user's input file, SCALE's functional modules BONAMI and NITAWL are run to extract data from a user-selected master cross-section library, to calculate resonance shielding corrections and other adjustments, and to then generate an AMPX Working Format cross-section library. Note that the user should select one of the coupled neutron/gamma libraries since total dose is the radiation quantity of interest.

(1) After reading in the user-specified radial sequence of material mix numbers and associated outer radii defining layers comprising the spherical reactor radiation source, the SCALE functional module XSDRNPM is run to perform a fission criticality calculation. Regardless of the criticality eigenvalue found, the spectral angular flux function of energy and direction as calculated on its outer spherical surface at radius $r=r_s$is recorded for later use.

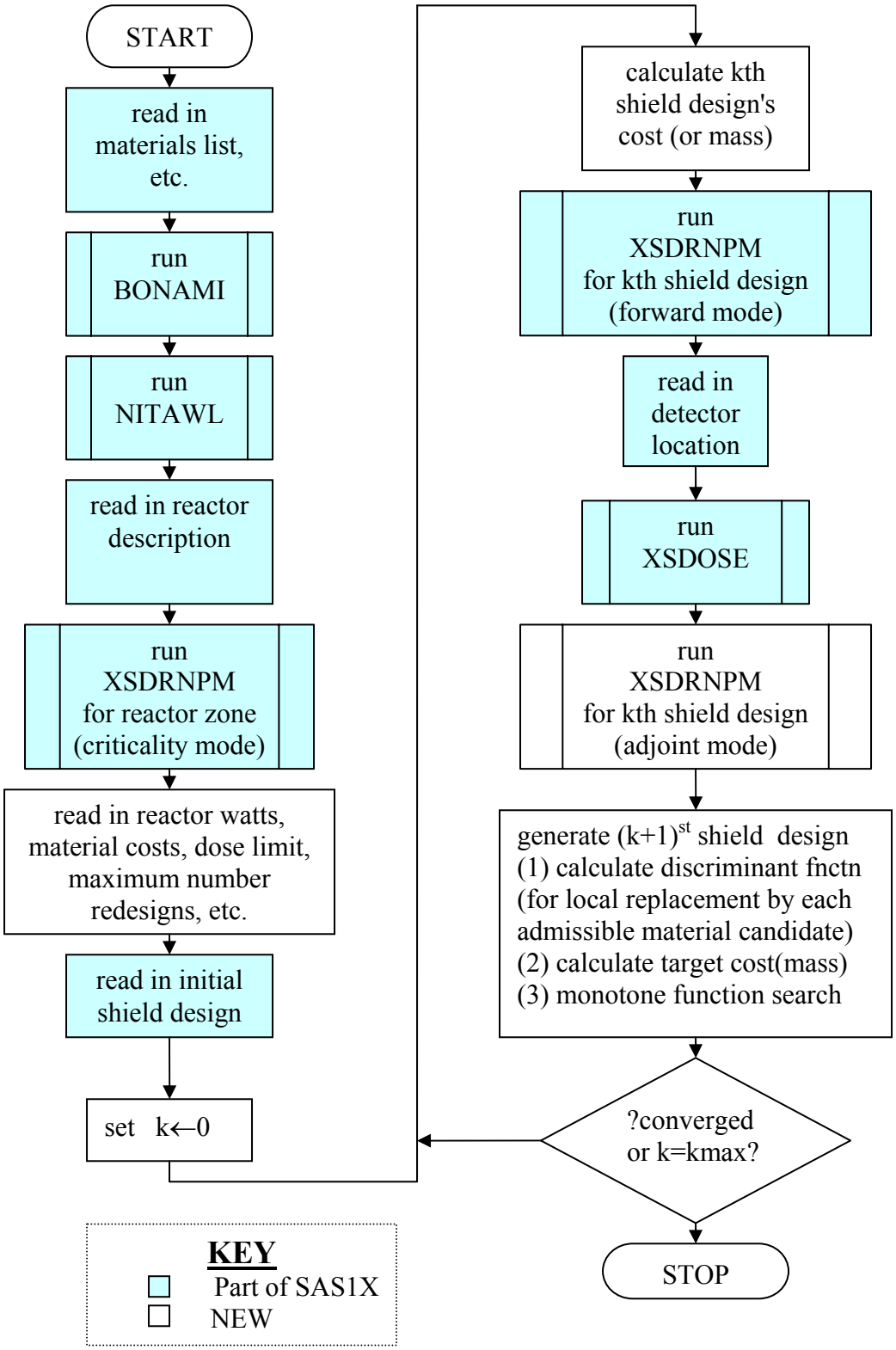


Figure 8-6: SAS1XOPT: A New Scale Control Sequence & Control Module for 1D Spherical Shield Optimization

[The assumption in this first XSDRNPM run of vacuum boundary conditions at radius $r=r_{source}$ was part of the SAS1X sequence, in which it was used to facilitate analyses of multiple user-defined radiation shield configurations within a single SAS1X run. It is retained in SAS1XOPT in order to allow results from this single XSDRNPM criticality run to be similarly applied to multiple *automatically redesigned* shields within a single SAS1XOPT run. The small error this approximation introduces by ignoring backscattering from the unmodeled shield could be estimated for the final shield redesign by appending a single final XSDRNPM run modeling the reactor and shield together.]

(2) The shield design iterations start with an initial provisional shield design specified by the user, defined by listing its radial sequence of layer material mix numbers and associated outer radii, as in the SAS1X sequence. Any such shield specification could alternatively be described in the mathematical notation introduced earlier as a particular m-vector function:

$$\underline{u}(r) \equiv \begin{pmatrix} u_1(r) \\ u_2(r) \\ \vdots \\ u_m(r) \end{pmatrix}; \quad r_{source} > r > r_{shield\_ext};$$

where $u_i(r)$ is the volume fraction at radial location r of shield material number i in a numbered list of m admissible shield materials. The domain of this function is $r_{source} > r > r_{shield\_ext}$, and its range is the following set of m-vectors:

$$\underline{u} \in U = \left\{ \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{pmatrix} : \quad 0 \le u_i \le 1 \quad \text{and} \quad \sum_{i=1}^{m} u_i \le 1 \right\}$$

The user-specified initial shield design can be denoted here as shield design number zero, i.e., ${}^{[0]}\underline{u}(r); \quad r_{source} > r > r_{shield\_ext}$, where the square-bracketed zero appearing in the pre-superscript position of this notation is the shield redesign iteration number. This design may contain an outer void layer (i.e., material number 0), since the shield region's user-input outer edge, r_{shield_ext}, is used subsequently within SAS1XOPT to restrict the outer surface of shield redesigns.

Note that the initial shield design does not need to be optimal, nor does it need to satisfy the user-specified requirements for radiation dose attenuation. However, iterative convergence of the shield redesigns may be accelerated if the initial provisional design is not far from optimal.

(3) For each provisional shield design in turn, starting with k=0 and repeating possibly up to the user-specified kmax, the algorithm calculates the following:

(a) The cost of this provisional design, ${}^{[k]}C$, is calculated using the mass densities of its materials and user-supplied costs per unit mass (which default to unit cost values for which this cost function equals total shield mass).

(b)The angular flux, ${}^{[k]}\psi\left(r,\hat{\Omega},E\right)$, is calculated throughout the shield region, $r_{source} > r > r_{shield\_ext}$, by using the SCALE functional module XSDRNPM to solve in 1D spherical geometry the source-free (forward) Boltzmann equation:

$$\hat{\Omega}\bullet\nabla\left({}^{[k]}\psi\left(r,\hat{\Omega},E\right)\right)+\underline{\sigma_t}^{T}(E)\left({}^{[k]}\underline{u}(r)\right)\left({}^{[k]}\psi\left(r,\hat{\Omega},E\right)\right)$$
$$-\oint\!\!\!\!\!\!\int d\Omega'\int_0^{\infty}dE'\underline{\sigma_s}^{T}\left(\hat{\Omega}'\rightarrow\hat{\Omega},E'\rightarrow E\right)\left({}^{[k]}\underline{u}(r)\right)\left({}^{[k]}\psi\left(r,\hat{\Omega}',E'\right)\right)=0$$

Here, the total and scattering cross section vectors for the *m* different admissible materials are denoted as follows:

$$\underline{\sigma_t}(E)\equiv\begin{pmatrix}\sigma_t^{(1)}(E)\\ \sigma_t^{(2)}(E)\\ \vdots\\ \sigma_t^{(m)}(E)\end{pmatrix};\qquad \underline{\sigma_s}\left(\hat{\Omega}'\rightarrow\hat{\Omega},E'\rightarrow E\right)\equiv\begin{pmatrix}\sigma_s^{(1)}\left(\hat{\Omega}'\rightarrow\hat{\Omega},E'\rightarrow E\right)\\ \sigma_s^{(2)}\left(\hat{\Omega}'\rightarrow\hat{\Omega},E'\rightarrow E\right)\\ \vdots\\ \sigma_s^{(m)}\left(\hat{\Omega}'\rightarrow\hat{\Omega},E'\rightarrow E\right)\end{pmatrix}$$

Vacuum boundary conditions are assumed on the outer surface of the shield and the angular flux from the central radiation source previously calculated in step (1) is applied to the shield's inner surface at $r=r_{source}$ after first scaling it up to the reactor thermal power level in watts specified by the user. That scale-up, which involves both the average number of neutrons per fission from the initial XSDRNPM run and also the energy released by each fission from the mix of fissioning isotopes, is automatically calculated by the SAS1XOPT code based on data extracted from the nuclide library and from the initial XSDRNPM run's output files.

(c) The dose rate, ${}^{[k]}D$, at the specified detector location in the external vacuum region, is calculated at $r = r_{detector} \geq r_{shield\_ext}$, by geometrically projecting the angular flux calculated on the shield's external surface in step (3a) and then applying energy-dependent ANSI standard flux-to-dose factors for humans, *w(E)*. This calculation is accomplished by the SCALE functional module, XSDOSE.

(d) The adjoint angular flux, ${}^{[k]}\psi^{*}\left(\vec{r},\hat{\Omega}',E'\right)$, is calculated throughout the shield region by using the SCALE functional module, XSDRNPM, to solve in 1D spherical geometry the source-free adjoint Boltzmann equation:

$$-\hat{\Omega}\bullet\nabla\left({}^{[k]}\psi^{*}\left(r,\hat{\Omega},E\right)\right)+\underline{\sigma_t}^{T}\left({}^{[k]}\underline{u}(r)\right)\left({}^{[k]}\psi^{*}\left(r,\hat{\Omega},E\right)\right)$$
$$-\oiint d\Omega'\int_0^{\infty}dE'\underline{\sigma_s}^{T}\left(\hat{\Omega}\rightarrow\hat{\Omega}',E\rightarrow E'\right)\left({}^{[k]}\underline{u}(r)\right)\left({}^{[k]}\psi^{*}\left(r,\hat{\Omega}',E'\right)\right)=0$$

together with the external boundary condition, $\psi^{*}\left(r_{shield\_ext},\hat{\Omega},E\right)=w(E)$

(e) Calculate the following 1D spherical "discriminant" m-vector function throughout the shield region:

$${}^{[k]}\underline{\boldsymbol{d}}\left(r;\,{}^{[k]}\psi^{*},{}^{[k]}\psi\right)=\oiint d\Omega\int_0^{\infty}dE\left({}^{[k]}\psi^{*}\left(r,\hat{\Omega},E\right)\right)*$$
$$\begin{pmatrix}-\underline{\sigma_t}^{T}(E)\left({}^{[k]}\psi\left(r,\hat{\Omega},E\right)\right)\\+\oiint d\Omega'\int_0^{\infty}dE'\underline{\sigma_s}^{T}\left(\hat{\Omega}'\rightarrow\hat{\Omega},E'\rightarrow E\right)\left({}^{[k]}\psi\left(r,\hat{\Omega}',E'\right)\right)\end{pmatrix}$$

SCALE did not previously have any functional modules to perform this calculation, so this calculation has been newly implemented. The discriminant m-vector functions of "r" are then multiplied by the Kuhn-Tucker parameter, τ, and the cost per unit volume of each of the m materials is added, i.e.,

${}^{[k]}\underline{b}(r)=\tau\,{}^{[k]}\underline{\boldsymbol{d}}(r)+\underline{c}$

(f) The m-dimensional control m-vector of material volume fractions in the next shield design, ${}^{(k+1)}\underline{u}(r)$, is adjusted to reduce the inner product $\left({}^{[k]}\underline{b}(r)\right)^{T}\left({}^{(k+1)}\underline{u}(r)\right)$ to less than the value of $\left({}^{[k]}\underline{b}(r)\right)^{T}\left({}^{(k)}\underline{u}(r)\right)$.. This adjustment within each iteration must be limited in order to obtain convergence. Two different alternative schemes were implemented for this adjustment, with the second proving to be much more reliable than the first.

(g) The algorithm exits the iteration loop if either a user-set maximum loop count is reached or if both the kth design's let-through radiation dose matches (within a small tolerance) the user-specified requirement and also the (k+1)st design matches the kth design. Otherwise the iteration continues, incrementing *k* and returning to step (3a).

8.3.3 Overall General Comments on Project

The project to develop a new SCALE control module for 1D spherical shield optimization was considerably more difficult to carry out than originally expected. It was not difficult to evaluate options for the use of existing SCALE modules, since the SCALE manual provides excellent documentation for users. New calculations not performed by existing SCALE modules needed to be coded and debugged. Difficulties here in translating between the continuous integrals developed herein and SCALE's discretized calculations using discrete energy and angle groups and Legendre flux

moments required care. For example, in spherical geometry with the energy spectrum replaced by discrete energy groups, the discriminant vectors as defined herein as :

$$\underline{\boldsymbol{d}}(r) = \oiint d\Omega \int_0^\infty dE\, \psi \left(-\underline{\sigma}_t \psi^* + \oiint d\Omega' \int_0^\infty dE' \underline{\sigma}_s \left(\hat{\Omega} \to \hat{\Omega}', E \to E' \right) \psi^* \left(\vec{r}, \hat{\Omega}', E' \right) \right)$$

needed to be replaced by discrete quadrature sums at each location:

$$\underline{d} = \sum_{g=1}^{G} \sum_{n=1}^{N} \frac{w_n}{2} \psi_{n,g} \left(-\psi^*_{n,g} \underline{\sigma}_g + \sum_{\ell=0}^{\infty} P_\ell(\mu_n) \sum_{g'=1}^{G} \underline{\sigma}_{\ell,g' \leftarrow g}\, \phi^*_{\ell,g'} \right)$$

where the Legendre flux moments are calculated from angular flux via the following quadrature:

$$\phi^*_{\ell,g'} = \sum_{n'=1}^{N} \frac{w_{n'}}{2} P_\ell(\mu_{n'}) \psi^*_{n',g'}$$

Combining these yields the actually coded forms to calculate the shielding discriminant vector directly from the angular flux variables of the discrete ordinates scheme.

$$\underline{\boldsymbol{d}}(r) = +\sum_{g=1}^{G} \sum_{n=1}^{N} \frac{w_n}{2} \psi_{n,g} \left(-\psi^*_{n,g} \underline{\sigma}_g + \sum_{\ell=0}^{\infty} P_\ell(\mu_n) \sum_{g'=1}^{G} \underline{\sigma}_{\ell,g' \leftarrow g} \sum_{n'=1}^{N} \frac{w_{n'}}{2} P_\ell(\mu_{n'}) \psi^*_{n',g'} \right)$$

$$= -\sum_{g=1}^{G} \sum_{n=1}^{N} \frac{w_n}{2} \underline{\sigma}_g \psi_{n,g} \psi^*_{n,g} + \sum_{\ell=0}^{\infty} \sum_{n=1}^{N} \sum_{n'=1}^{N} \frac{w_n w_{n'}}{4} P_\ell(\mu_n) P_\ell(\mu_{n'}) \sum_{g=1}^{G} \sum_{g'=1}^{G} \underline{\sigma}_{\ell,g' \leftarrow g}\, \psi_{n,g} \psi^*_{n',g'}$$

It was also time-consuming to puzzle out how to directly access the AMPX formatted working cross section library from new FORTRAN code in order to obtain information needed for the optimal design algorithms that was not already being accessed by existing SCALE modules. For example, the energy released from each fissioning isotope was needed to normalize radiation dose to the user-specified reactor power, and isotopic mass was needed to accurately shield mass and thus cost. However, it had been partially anticipated that these activities would take significant effort, since the SCALE manual does not contain notes for FORTRAN programmers trying to develop modifications.

However, by far the most difficult unanticipated impediments to completing this project were related to the SCALE system's overall software structure. SCALE is not a single FORTRAN program with the typical FORTRAN subroutine-calling interfaces between separate calculational procedures. Instead, it is a collection of independent codes which, in conjunction with the computer's operating system, are caused to execute sequentially in batch mode. When a PC-based version of SCALE is executed to run one of its control sequences, what really happens is that a sequence of batch job requests is automatically generated and submitted through the Windows Shell to the Windows batch operating system. Each SCALE batch job terminates its execution under Windows before the next SCALE batch job is initiated by Windows. In many of the SCALE sequence cases, the actual sequence of batch job requests submitted to Windows is not generated all at once since later SCALE batch job requests may depend on the outcomes of earlier batch jobs. In addition, the SCALE control modules that generate these job requests for Windows are

themselves also batch jobs, and each short list of sequential batch jobs that a control module generates is usually ended by a request to rerun itself.

SCALE therefore relies heavily on semaphores and various special temporary files to achieve its functions. Although most of SCALE software is written in FORTRAN, some is written in the C++ language and SCALE also makes use of the Windows Shell programming language. SCALE includes a small executive program written in FORTRAN called the SCALE system driver which resides in memory throughout an entire user-initiated SCALE sequence of multiple batch programs. FORTRAN code in this system driver effectively creates, in conjunction with the Windows operating system and the execution of other SCALE modules, a sequential machine process which alternates between

- reading portions of a user-written input file,
- reading an "alias" file to check whether to redirect to a different control module,
- writing selected portions of the user-written input file to another file for control modules,
- updating a special communication file of status information for control modules to read,
- reading the updated communication file after a control module has altered it, and
- submitting batch requests to the Windows operating system (while suspending self execution until each request has been completed).

Although the SCALE manual contains a brief description of the SCALE driver, it fails to explain many of the parameters in the file it uses to communicate with control modules or with FORTRAN subroutines that the control modules call.

A consequence of SCALE's configuration structure as a set of independent codes operating separately within the Windows Shell's batch system is that communication between SCALE modules must rely on the Windows file system. If data obtained in one SCALE module by reading a user input file, by reading cross section files, or by making its own calculations, is to be available to another SCALE module executing later, then that data must be explicitly saved to a Windows file by FORTRAN code in the earlier executing module. Since a control module may typically need to execute and terminate execution multiple times in a single SCALE control sequence, it must similarly contain special FORTRAN code to talk to its later self. It must save to a file its internal data, including special programmer-defined flags to indicate which point in the control sequence it had gotten to before terminating so that it does not repeat itself .

Because of the complicated and delicate nature of the entire SCALE system there was an obvious concern that any modifications made to it by the present project might compromise it. It was therefore resolved that no changes whatsoever would be made to any existing SCALE functional or control modules. The new capability to automatically design spherical shields that are optimal in the sense of minimizing material cost would need to be produced by creating an entirely new control module code. The starting point for developing that new code would be a copy of the preexisting SAS1 control module code, and it would freely used code excerpts copied from other SCALE modules .

Because of the decision to not modify any existing functional modules, data calculated by a functional module at an earlier part of the SAS1XOPT sequence but not saved to a file would need to be recalculated as part of the new SAS1XOPT module's job. This situation occurred several times, particularly for information calculated by the XSDRNPM module and by material information processor subroutines. There may also have been situations where data had been saved to a file but the lack of documentation concerning those files led to ignorance about its availability and to recalculating it within newly coded subroutines of SAS1XOPT.

Much of the unanticipated time-consuming effort required for this project was devoted to "reverse-engineering" SCALE's existing FORTRAN code in order to figure out how to properly interface with it. No separate document was found explaining its internal algorithms. Textual comments in the SCALE code are both terse and sparse. Even the assignment of FORTRAN variable names to its calculated internal quantities is sometimes inconsistent between different subroutines. Variable names also frequently use non-semantic character combinations which fail to suggest any physical meaning. In spite of these difficulties, the project was finally completed. The new SAS1XOPT control module does successfully design sequences of shields and the sequences it generates do appear to be converging on optimal designs.

8.4 Running SCALE's new SAS1XOPT control sequence

Before running on a PC the new SAS1XOPT SCALE control module developed herein for automatic optimization of 1D spherical shields, it is necessary to first copy the executable file resulting from its compilation and linking, SAS1XOPT.EXE, into the scale5 subfolder named bin.

To run any scale5 option, one prepares and names an appropriate text file to serve as input to the scale5 system. This text file's extension must be in . The batch shell window is then opened and one types scale5 followed after a separating space by the name of the prepared text file, not including its extension.

The scale5 system expects certain pattern rules to be followed in prepared text files, and individual control sequences have additional rules. The SAS1XOPT input file follows the format set up for the SAS1X sequence everywhere but in one section, which was newly created to provide additional input data needed for optimization. The other input file contents are documented in the SCALE5 manual, so are only referenced briefly here.

The input text file is line oriented. Consecutive input lines with a common purpose form sections. These are listed in their required sequential order in the following table, which delineates in boldface type the new input data added to the SAS1X input data for the SAS1XOPT control sequence. Comments may appear after the first line, identified by an apostrophe in the first column position. They do not affect SAS1XOPT execution.

Tables 8-1 and 8-2 list the new input parameters for SAS1XOPT and defaults. Each input parameter added to control shield optimization is optional because each is defaulted in the code to predefined values. It is only necessary to input any of them if the user wants it to take a different value. If input, each should be given its own line.

8.4.1 Observed performance of the algorithms.

Initial runs, in which the full change indicated by the iterative extension of Pontryagin was implemented in each iteration, spectacularly failed to converge. Successive designs would suddenly insert air layers throughout the shield, or fill the entire available space with heavy metal. The first algorithm proposed to avoid this behavior censored the changes by only implementing a user-specified fraction of the shielding layer thicknesses having the largest local improvement in the inner product. This usually worked if that fraction was set to less than about 0.01, which resulted in very slow improvements. However, it also tended to get stuck in oscillations between a pair of off-optimal designs differing only by a single layer. When stuck in this manner, intervention by further reducing the input fraction of total thickness that could be changed would get it unstuck.

The second algorithm has worked quite well, with little pathological behavior. In order to use the available discrete ordinates code directly, it was necessary to divide the shield thickness into discrete "regions" so that within each region a mixture of materials is approximately implemented as adjacent pure layers. During each iteration this algorithm adjusts the average m-vector, u, within each region so that the inner product between it and the average b in that region is minimized, subject to the normal constraints on u and subject also to the constraint that the u not change from its previous m-vector value by a distance more than the input parameter, p04. With the input parameter set to 0.05 this has worked well.

Table 8-1: Contents of Input File For Running SAS1XOPT

Line Content	Purpose	Required/ Optional	Number of lines
#sas1xopt (& optional parm spec.)	control sequence name	required	1
(any phrase)	reactor criticality run title	required	1
27n-18couple multiregion	lib. id/xsec treatment option	required	1
(mixture descriptions)	identify materials in list	required	several
end comp	signals end of mix descr.	required	1
spherical vacuum end	reactor criticality options	required	1
(mix id numbers vs. radii)	reactor geometry description	required	several
end zone	signals end geometry input	required	1
more data isn=16 end	matches angular quadratures	required	1
end	signals no more reactor data	required	1
last	left over from the SAS1X logic	required	1
read opt	**signals start of optimiz. data**	**required**	**1**
(keywords and numerical values)	**shield redesign control data**	**optional**	**several**
end opt	**signals no more optimiz. data**	**required**	**1**
(phrase)	title for shielding runs	required	1
spherical	geometry option	required	1
(mix id numbers vs radii & flags)	geometry descr. initial shield	required	several
end zone	signals end geometry input	required	1
ndetec=1	sets number of external doserate evaluation locations	required	1
read xsdose	signals start of XSDOSE input	required	1
(a number)	radial location of external doserate evaluation location	required	1
end	signals end of input data	required	1

Table 8-2: New Defaulted Input Parameters Added for SAS1XOPT

keyword input	default value	meaning
pwr=	1	reactor thermal power (watts)
dsr=	1	doserate limit (rem/hr)
mzo=	100	max # of material zones in shield
alg	2	Algorithm switch (1 or 2)
mxi=	500	max # of shield integration intervals
mns=	0.1	min. size of shield integration interval
itr=	5	max redesign loop iteration count
p1=	0.2	Doserate exponent, alg2
p2=	0.1	Not used
p3=	0.1	Not used
p4=	0..05	Max L2 measure of step size in u-space
p5=	0.1	Not used
cst(#)=	50	array by material ID# of shield material prices per unit mass on Mars, ($/gram)

CHAPTER 9
9 FAMILIES OF OPTIMAL SPHERICAL SHIELDS

Shields for a family of mobile nuclear engines with different power levels was examined using the new SAS1XOPT scale module developed as part of this thesis work, in order to investigate optimal shields of different sizes. The selection of reactor thermal power for each engine was made as part of the work on thermal conversion systems described in a later chapter. However, its results are listed in the following table along with other summary details. All these shield designs assumed a spherical reactor 8.77 cm in radius surrounded by a tungsten neutron reflector extending from the reactor out to a radius of 23.9255 cm. The size parameters had previously been shown appropriate for criticality and for heat transfer, at least for a 1 MW thermal reactor. The shielding to be designed was defined to SAS1XOPT as starting at a radius of 23,8255 cm and potentially extending out to a radius of 200 cm. For each case, SAS1XOPT was instructed to pursue optimal shield designs with a total leakage dose rate of 1.375 mR/hr at R=600 cm.

Tables 9-1 and 9-2 summarize aspects of the resuts for each of the nine designs having differing power levels. Table 9-1 lists each reactor power level, leakage radiation dose, reactor plus shield mass, and outer shield radius. Of the 12 shielding materials available for use by the SAS1XOPT code, the code chose to use only four as listed in Table 9-2.

SAS1XOPT preferentially selected only tungsten, boron-10, 6LiH and U-238 for use. Indeed, whenever the program was started with an initial shield design containing other materials, e.g., lead, plastic, beryllium, graphite, etc., they were quickly eliminated by the algorithm within a few redesign iterations.

To present the material arrangements chosen by the SAS1XOPT module for the 9 different shields, the Figures 9-1 through 9-9 were prepared using MATLAB graphics. Different colors represent the different shielding materials used, and the ratios of their areas represent relative mixing fractions. Interesting observations include facts that
(1) the optimization ended up with large regions of pure material, and
(2) the higher attenuation accompanying higher reactor power is accomplished by SAS1XOPT mostly by changing the optimal mix of ingredients, without greatly increasing the shield's outer radius

Table 9-1: Parameters of Optimal Shield Designs Found by SAS1XOPT

#	Shaft Power (hp)	Reactor Thermal Power(MW)	Final Kuhn-Tucker Parm (τ)	Reactor+ Shield Mass (Tonnes)	Shield Outer Radius (cm)	Neutron+Gamma TotalDoserate at R=600 cm (mr/hr)
1	100	0.310	18139	18.949	133.14	1.367
2	178	0.4958	22230	20.220	133.51	1.354
3	316	0.8217	26654	21.788	134.44	1.353
4	562	1.368	31813	23.676	135.09	1.349
5	1000	2.291	44506	26.154	135.62	1.304
6	1780	3.931	60056	29.320	136.15	1.294
7	3160	6.879	86756	34.178	136.77	1.222
8	5620	12.136	101342	38.219	137.59	1.374
9	10000	21.274	131480	45.671	138.43	1.323

Table 9-2: Admissible Shielding Materials and their use by SAS1XOPT

Tungsten	Used
Boron-10	Used
Beryllium	Not used
Iron	Not used
Polyethylene plastic	Not used
Lead	Not used
6Lithium Hydride	Used
Uranium-238	Used
Graphite carbon	Not used
7Lithium Hydride	Not used
Boron-10 Carbide	Not used
Water	Not used

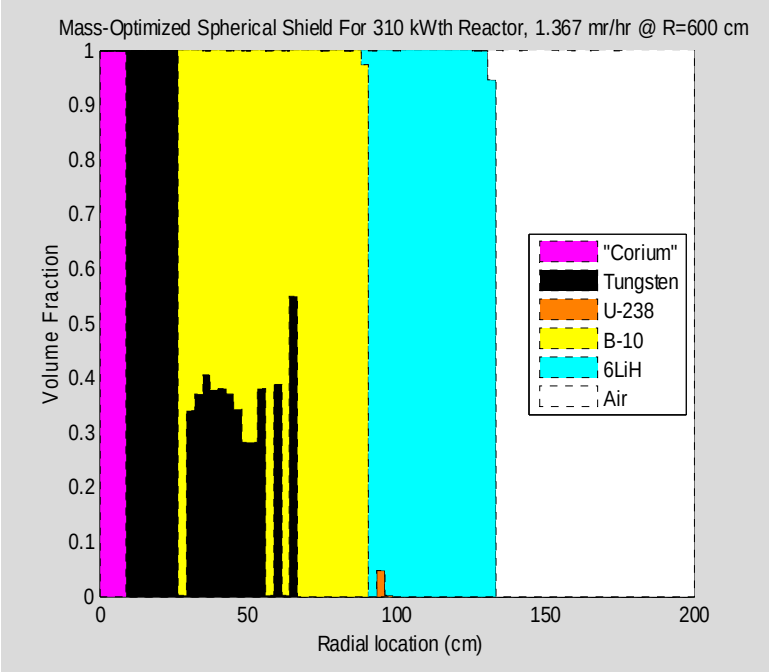


Figure 9-1: Reactor and Shield Material Mixes in Mass-Optimized Design #1

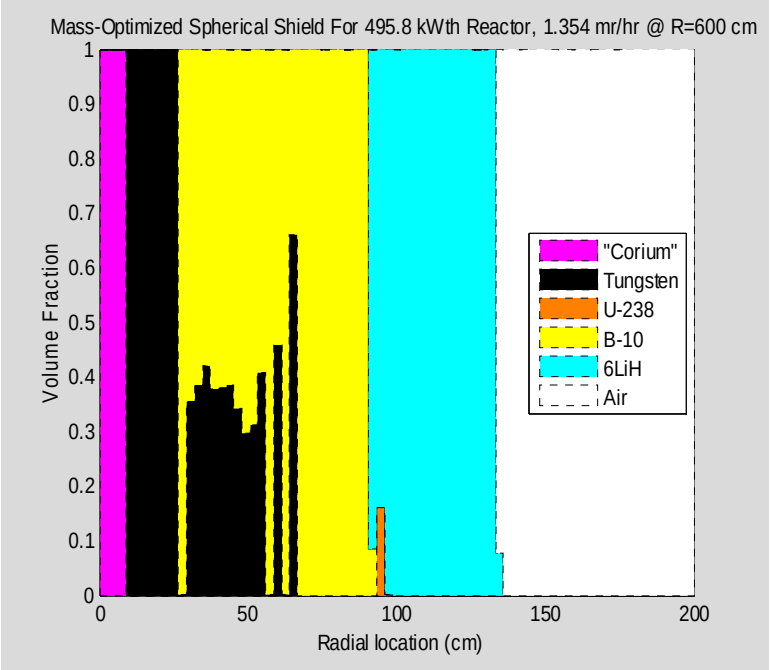


Figure 9-2: Reactor and Shield Material Mixes in Mass-Optimized Design #2

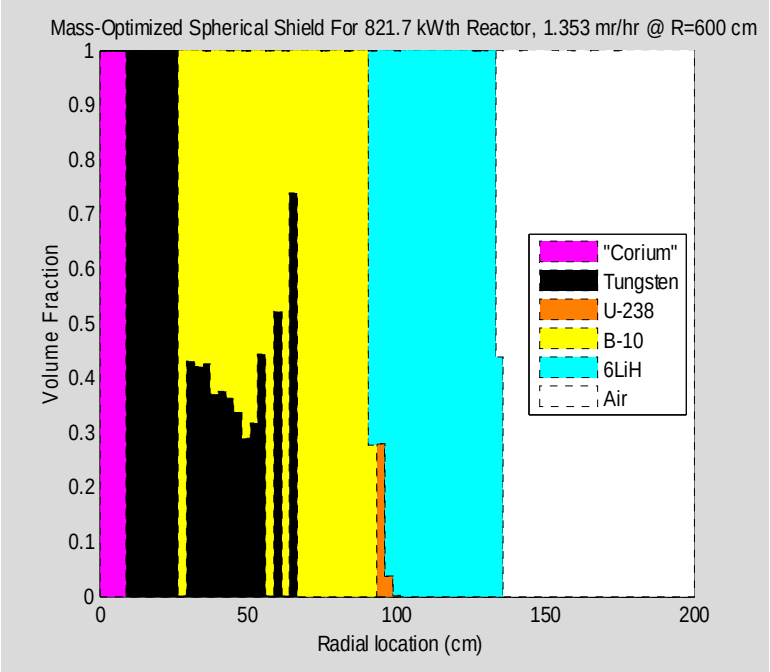


Figure 9-3: Reactor and Shield Material Mixes in Mass-Optimized Design #3

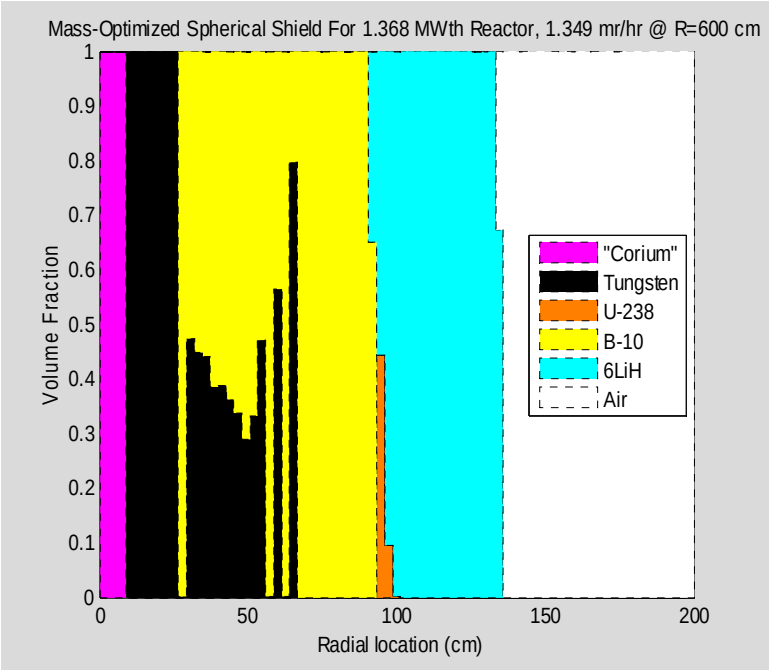


Figure 9-4: Reactor and Shield Material Mixes in Mass-Optimized Design #4

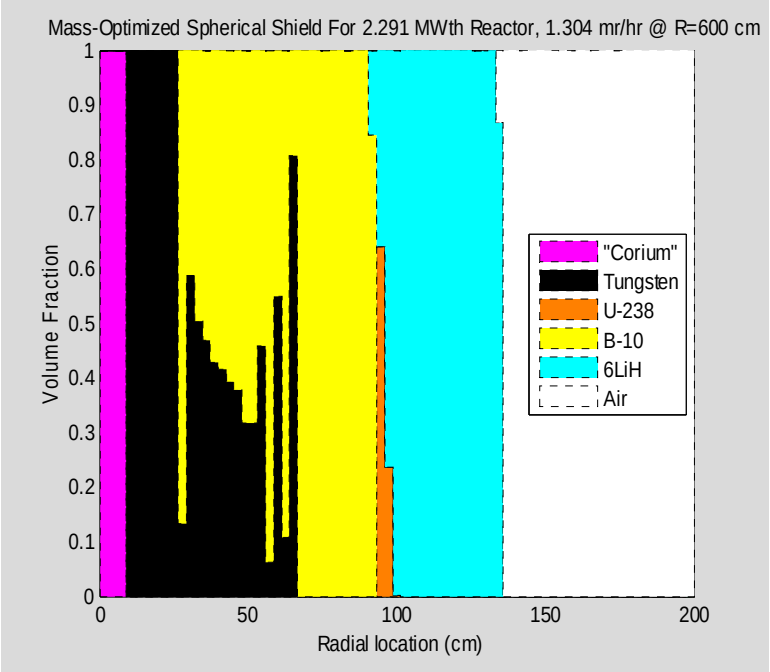


Figure 9-5: Reactor and Shield Material Mixes in Mass-Optimized Design #5

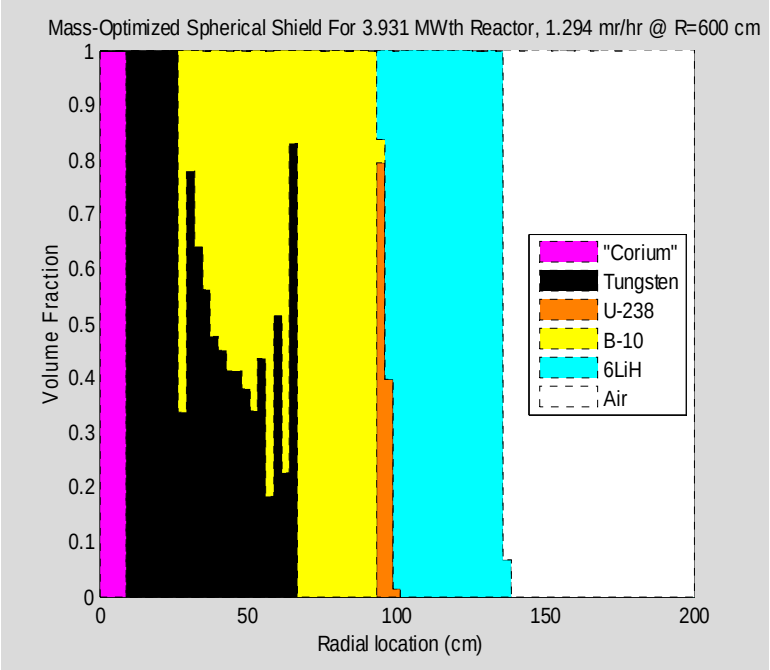


Figure 9-6: Reactor and Shield Material Mixes in Mass-Optimized Design #6

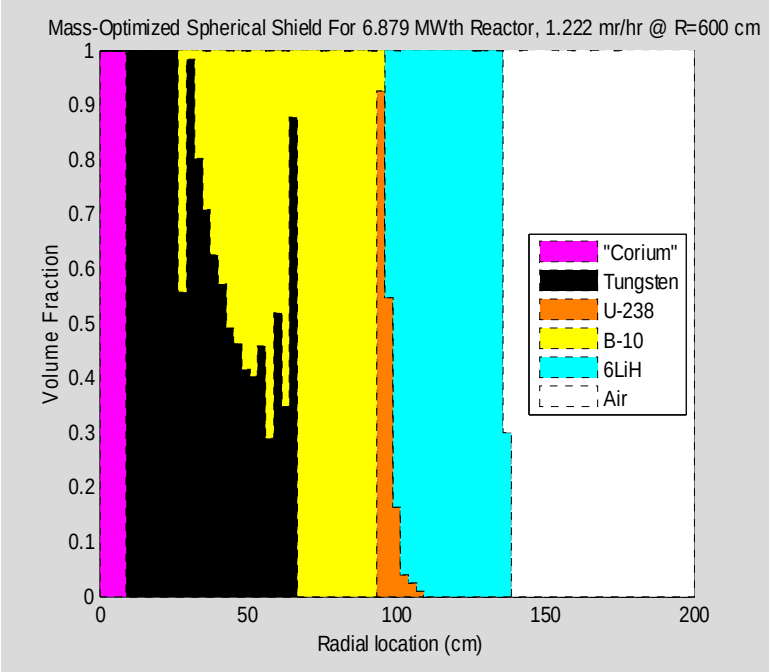


Figure 9-7: Reactor and Shield Material Mixes in Mass-Optimized Design #7

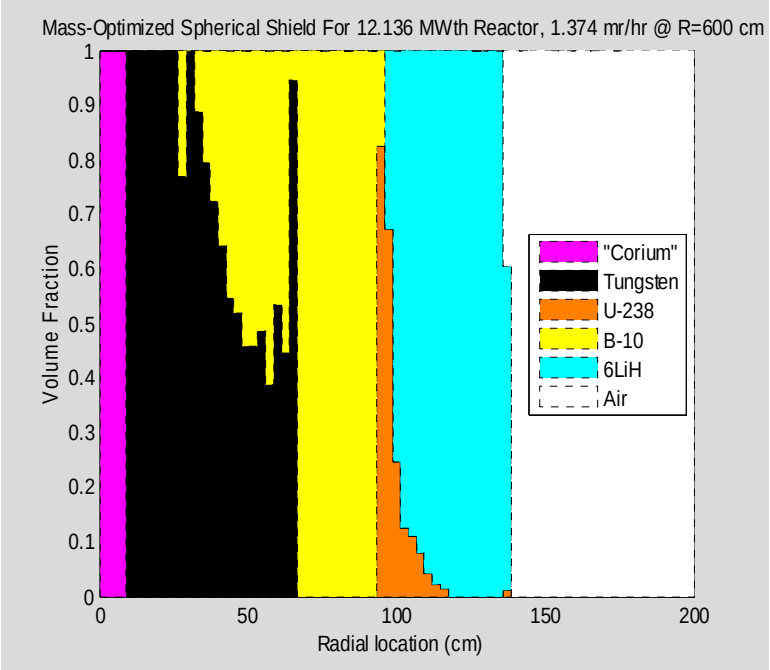


Figure 9-8: Reactor and Shield Material Mixes in Mass-Optimized Design #8

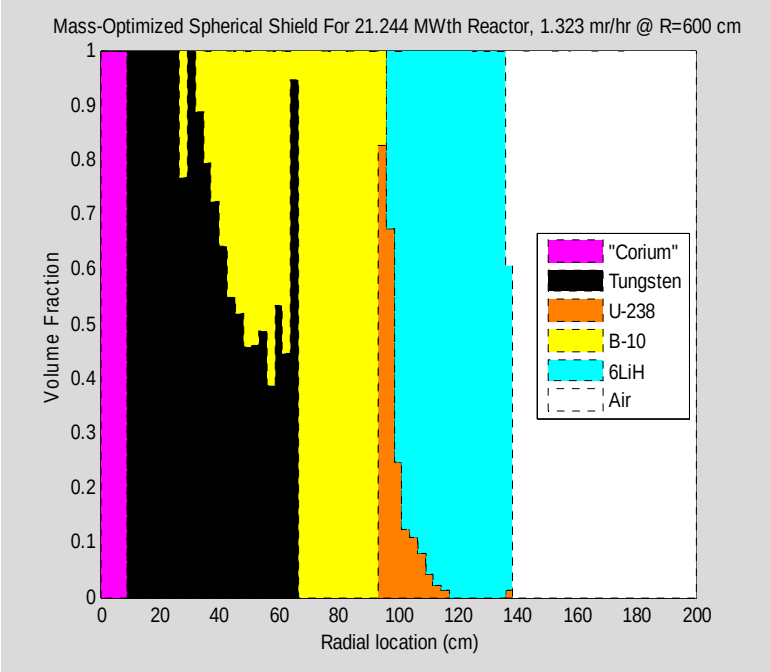


Figure 9-9: Reactor and Shield Material Mixes in Mass-Optimized Design #9

There is a remaining issue challenging these sizing results. Each of these nine shielding designs would provide the attenuation ratio reported in Table 9-1 so that the leakage doserate at R=6 meters would indeed be the reported fractions of reactor thermal power. However, some of the higher-powered designs may not have allowed enough space for the reactor to transfer its heat to the reactor coolant within feasible heat flux limitations. The reactor radius of 8.77 centimeters was selected based on analyses of criticality and the requirement for heat transfer at a reactor thermal power of 1 MW. For designs #1 through #3 which have lower thermal power levels, this minimal radius is necessary for criticality and heat transfer issues are not limiting. Design #4 which must transfer 1.368 MW of heat may perhaps have some difficulty with heat transfer issues if its reactor is limited to an 8.77 cm radius sphere. Certainly design #9 with its 21.244 MW thermal power would suffer from heat transfer inadequacy. Thus, the higher power designs should be allocated larger volumes. If we choose the volume per thermal watt to match the ratio for the 1 MW 8.77 cm case, the conclusion follows that design # 9 should have a spherical reactor core 24.3 cm in radius.

A SAS1XOPT run requiring that radius surrounded by a tungsten neutron reflector with the same thickness as before found a high criticality eigenvalue (k_{eff}=1.7) but more significantly resulted in the SAS1XOPT shield design of Figure 9-10. Its performance is summarized in Table 9-3. It may seem surprising that the mass increase to accommodate the larger reactor core was less than one tonne.

Yet another SAS1XOPT run was made to illustrate a shield design in which unequal costs per unit mass were assigned to different materials. Design #3 from Table 9-1 and Figure 9-3 was chosen as this design's starting point. However, in this case, water was assigned a cost less than one peney per gram, while all other shielding materials were assigned costs of $50 per gram. This actually represents a realistic situation since water ice is abundant on Mars and will be mined during manned missions in order to make rocket fuel for the return trip back to Earth. Additional water could thus be available for radiation shielding without paying the large cost of transporting it from Earth.

The spherical shield design resulting from this SAS1XOPT run attenuates the radiation from a 0.8217 MWt reactor to a total neutron plus gamma ray doserate of 1.38 millirem per hour 6 meters from the reactor's center. All materials transported from Earth in the reactor and its shielding together have a mass of 17.85 tonnes, but another 4.80 tonnes of martian water is also incorporated into the shield. Thus, the total mass of reactor plus shield is 22.65 tonnes. This exceeds the design #3 mass as listed in Table 9-1, but the cost of transportation from earth is less than the cost of transporting design #3.. The material mix structure of this cost-optimized reactor plus shield design incorporating cheap martian water is illustrated in Figure 9-11.

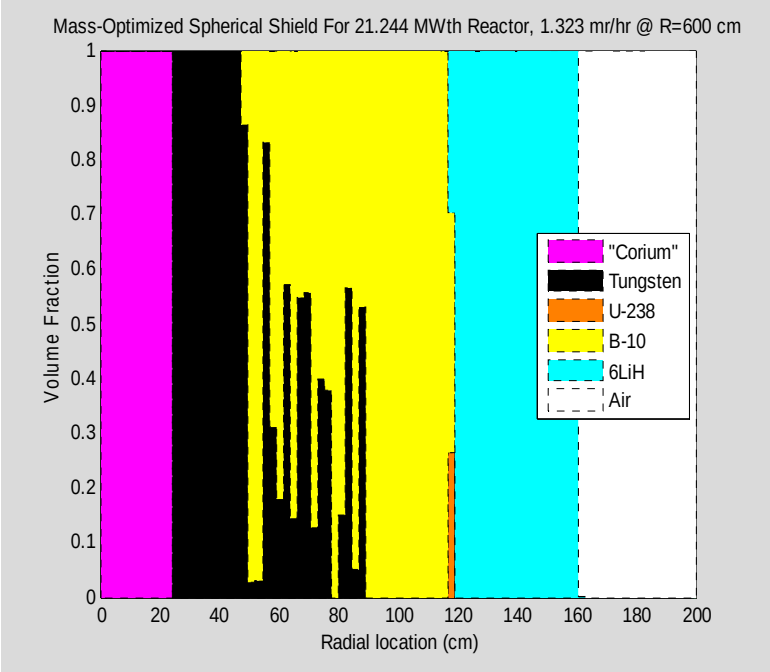


Figure 9-10: Reactor and Shield Material Mixes in Mass-Optimized Design #9 Allowing Space for a 24.3 cm Radius Spherical Reactor

Table 9-3: Spherical Shields For R=8.77 cm vs R=24.3 cm Radius Reactors

#	Shaft Power (hp)	Reactor Thermal Power(MW)	Final Kuhn-Tucker Parm (τ)	Reactor+ Shield Mass (Tonnes)	Shield Outer Radius (cm)	Neutron+Gamma TotalDoserate at R=600 cm (mr/hr)
9	**10000**	**21.274**	**131480**	**45.671**	**138.43**	**1.323**
10	**10000**	**21.274**	**29928**	**46.149**	**160.71**	**1.377**

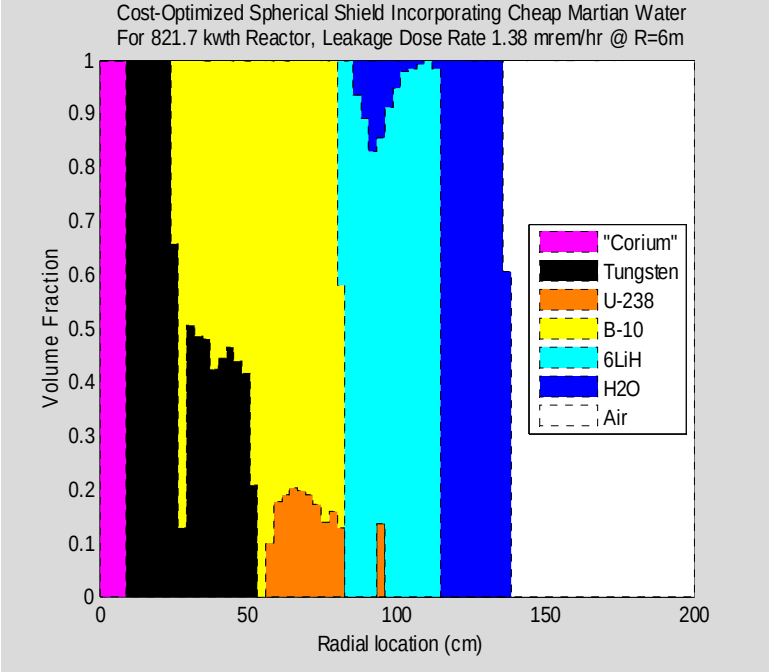


Figure 9-11: Illustration of Cost-Optimized Shield Incorporating Materials With Different Costs Per Unit Mass

CHAPTER 10
10 FUTURE 3D SHIELDING STUDIES

It is useful in early design activities such as the present scoping studies to use 1D codes to inexpensively investigate broad shielding issues. Only 1D analyses have been conducted so far in the present effort. However, if nuclear engines for Mars will be developed, then as additional design details are tackled this will becomes inadequate, and 3D radiation analyses will become essential.

For nuclear applications to manned Mars missions where the costs of transporting shielding to Mars are significant and where there are no members of the background public requiring radiation protection, it is appealing to consider unconventional designs in which directional shields protect mission members while intense radiation is allowed to spill out in unoccupied directions. For instance, one could even imagine moveable shields that automatically remain between a reactor and a nearby suited astronaut. Although significant costs might be saved by such novel active approaches, much 3D radiation transport analysis effort would need to be done to assure astronaut safety. A more conventional approach to cost savings would be to simply provide less shield thickness on an engine's top and bottom under the plausible assumption that astronauts will not occupy those locations during high reactor power operations. However, the need to choose how much to shave off the shielding there requires 3D predictions of the backscatter from the ground under the engine and from the air above it. For long-distance mobility in a RV-like ground vehicle, some shielding would be provided by the placement of supplies and equipment between the astronauts and the reactor-engine. To understand the effect of such arrangement options requires 3D radiation transport analysis.

The difficulty of 3D transport studies is also their strength. Although each run provides an enormous amount of detail on the radiation field, it also requires that an enormous amount of detail about the physical configuration must first be supplied. Early in a design, most of the fine geometric details have not yet been decided. Thus, 3D radiation studies should ideally be done in conjunction with the decisions developing a design.

One important set of 3D radiation transport analyses to do for the development of nuclear engines of the present concept is to synthesize a geometrical arrangement of shield penetrations for liquid lithium flow as it circulates between the reactor and an external heat exchanger transferring heat into the compressed martian air. Both neutrons and gamma rays pass essentially unimpeded through liquid lithium, so the flow passages should be given some sort of labyrinthine shape to avoid compromising the shield.

CHAPTER 11
11 SENSITIVITIES OF REACTOR & SHIELD DESIGNS

Many design decisions must be made with the reactor and shield and also with the thermal conversion system before the engine design is fully complete. However, many of them have multiple effects to the entire system. In order to make intelligent over-all decisions concerning an unfamiliar design it is conventional to develop an integrated approximate model of the interrelationships. For instance, choosing a higher reactor coolant flow rate reduces the temperature difference it needs to carry a given thermal power from the reactor. If the maximum coolant temperature is limited this increases the coolant's average temperature which in turn increases thermal conversion efficiency so that less reactor power is needed for the same engine power output. This slightly reduces the needed mass of the radiation shield. On the other hand, the larger coolant flow area increases radiation losses through the coolant channel and might require compensating increases in shielding mass.

Although establishing the interrelating design sensitivities and implementing a "systems code" to study them is an essential step, it has only started during the present effort. At this point it remains mostly a future activity remaining to accomplish.

CHAPTER 12
12 THERMAL CONVERSION SYSTEM FUNDAMENTALS

Thermal conversion systems, also known as engines, are comparatively new in humanity's approximately 50-century old written history. Four centuries ago they had not yet been imagined, three centuries ago they were in their infancy as schemes to pump water from mines, two centuries ago they were in rapid development. Engines have since powered mobile vehicles on the oceans, on the land, and through the air. Engine thermal conversion issues still form the central core of today's engineering disciplines. This chapter reviews thermal conversion generally, emphasizing issues such as waste heat rejection that are crucially important for mobile engines operating on Mars' surface.

12.1 Essential Theory

12.1.1 Thermal Conversion System Energy & Entropy Flow Diagram

Every thermal conversion system is abstractly represented by the Figure 12-1 flowchart:

12.1.2 Laws of Thermodynamics

The first two laws are used frequently with thermal conversion systems while the third law, thanks to Nernst, is useful in calculating properties of chemically reacting mixtures.
(1) **Total System Energy is Conserved**
(2) **Total System Entropy Never Decreases**
(3) **Substance Entropy Goes to Zero at Absolute Zero**

Energy flowing as heat, Q, anywhere in Figure 11-1 has an *absolute* local temperature, T. The associated entropy flow is the ratio of the heat flow to absolute temperature, i.e.,

$$S \equiv \frac{Q}{T} \tag{12-1}$$

Zero entropy flow accompanies the flow of energy as pure work.

For a thermal conversion process continuing indefinitely it is appropriate to assume the heat and entropy flows of Figure 12-1 all occur during some fixed time period and that conversion system states at the end of that period match the states at the beginning of that period. Since then there are no internal state changes to consider, the first law states that total energy outflow matches inflow, i.e.,

$$W + Q_{COLD} = Q_{HOT} \tag{12-2}$$

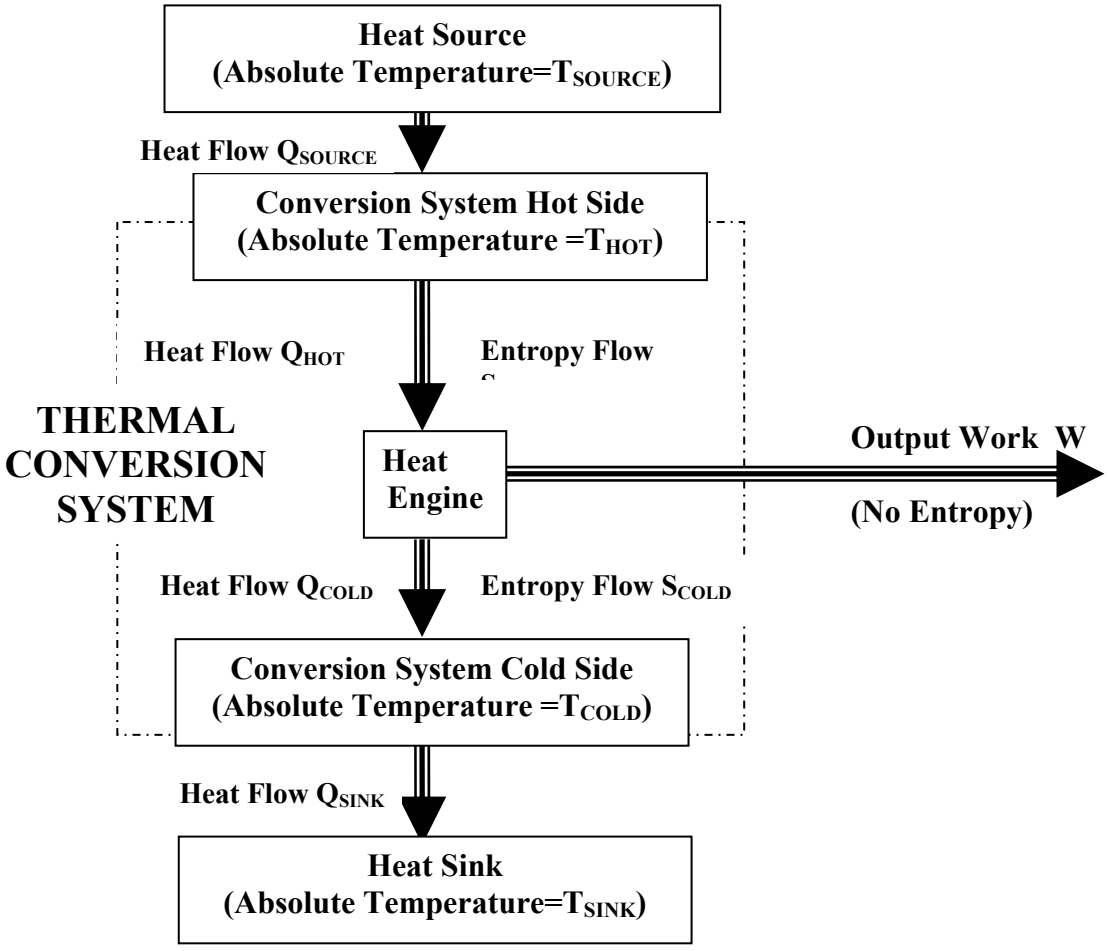


Figure 12-1: Energy and Entropy Flow in Thermal Conversion

With the same assumption the second law applied to the Heat Engine in Figure 12-1 states that its total entropy outflow is at least as large as its inflow.

$$S_{COLD} \geq S_{HOT} \Rightarrow \frac{Q_{COLD}}{T_{COLD}} \geq \frac{Q_{HOT}}{T_{HOT}} \qquad \text{(12-3a, 3b)}$$

Combining these statements yields the inequality limiting thermal conversion efficiency.

$$\varepsilon \equiv \frac{W}{Q_{HOT}} \leq 1 - \frac{T_{COLD}}{T_{HOT}} \qquad \text{(12-4)}$$

The first law applied to heat flows between the conversion system and its external heat source and sink states that these flows match the corresponding internal heat flows.

$$\begin{aligned} Q_{SOURCE} &= Q_{HOT} \\ Q_{COLD} &= Q_{SINK} \end{aligned} \qquad \text{(12-5a, 5b)}$$

The second law similarly applied implies simply that heat cannot externally flow from colder to hotter temperatures. Thus for positive values of thermal conversion work output, the following linked inequalities are always satisfied.

$$T_{SOURCE} \geq T_{HOT} > T_{COLD} \geq T_{SINK} \qquad \text{(12-6)}$$

Because of design tradeoffs in many implementations, a thermal conversion system's hot and cold side temperatures, T_{HOT} and T_{COLD}, may cover a significantly smaller range than covered by the heat source and sink temperatures, T_{SOURCE} and T_{SINK}. That reduces conversion efficiency in order to improve some other system performance measure (e.g., cost, size, or weight). An example of this is found in typical designs of the Pressurized Water Nuclear Reactor (PWR) and its steam engine thermal conversion system. Temperatures within the uranium dioxide fuel, corresponding to T_{SOURCE} in Figure 1, may reach 1000 Kelvins or more. However, the maximum PWR water/steam temperature, corresponding to T_{HOT} in Figure 12-1, is kept typically below 570 Kelvins in order to limit maximum system pressure and the associated pressure vessel costs.

12.2 The Waste Heat Rejection Bottleneck

A commonly underestimated thermal conversion system difficulty is in achieving adequate waste heat rejection. Although heat sources such as nuclear reactors may be configured as entirely internal portions of engine structures which also include thermal conversion systems, those **engines cannot be designed independent of their external environments**. All thermal conversion systems must reject waste heat to an external heat sink. This corresponds in Figure 12-1 to the heat flow Q_{COLD} being transferred from

T_{COLD} to T_{SINK}. Since no practical thermal conversion system energy efficiency has yet exceeded about 50%, since most efficiencies are in the 20% to 40 % range and since some are even lower, it follows that Q_{COLD}, the waste heat to reject, is not small. It is at least as large as the output work, W, it is typically 1.5 to 4 times as large as W, and in some cases it is even more.

The need for substantial external waste heat rejection poses an insurmountable problem for proposed applications in which appropriate external heat sinks do not exist. For instance, consider the planet, Venus, with its dense carbon dioxide atmosphere at a surface pressure 92 times Earth's sea level pressure and its resulting runaway greenhouse effect producing a surface temperature averaging 737 Kelvins. A PWR and its associated steam cycle thermal conversion system deployed remotely on the surface of Venus could never reject its waste heat at the 300 Kelvins temperature typically required for PWRs.

Assuming that an appropriate external heat sink exists for a thermal conversion system application, there can still be difficulties in coupling to it. Often, a need for designers to limit the cost, size, or mass of heat rejection equipment leads to selecting a cold side temperature, T_{COLD} in Figure 12-1, considerably higher than the external heat sink temperature, T_{SINK} in Figure 12-1. This is because heat transfer is simplified at larger temperature differences, although energy conversion efficiency suffers.

12.2.1 Convective Heat Rejection to Environmental Fluids

Most thermal conversion systems in operation today reject their waste heat to moving environmental fluid streams. Of the conventional implementations in common use, only one is appropriate for Mars.

12.2.1.1 Heat Rejection By Water Evaporation

Changing the phase of a substance from solid to liquid or from liquid to gas is accompanied by absorption of energy. Without any temperature increase, evaporating liquid water absorbs more than 500 times as much heat as would be absorbed in increasing its temperature by one degree Kelvin. Water's evaporative change in enthalpy at Earth's sea level pressure depends only slightly on its temperature, ranging from 2.51 Megajoules per kilogram at 0 C to 2.26 Megajoules per kilogram at 100 C. Thus, about 0.4 kilograms per second of water evaporation can dissipate one million watts of heat flow. For example, the 2500 MW of waste heat which must be rejected from a terrestrial 1.3 GWe PWR is typically absorbed by evaporating approximately 1000 kg of water per second in adjacent cooling towers. This requires a local makeup water supply of about 1 m^3/sec, which can be a trivial amount to extract from a sufficiently large nearby river. Evaporation removes the liquid water from the local vicinity of the power plant and injects it as gas into the air, but the water is eventually recycled as distant rain or snow.

Most applications of evaporative cooling are configured indirectly in two steps. In the first step, waste heat rejected from a thermal conversion system or some other heat producing process is transferred to a loop of circulating water. In the second step, that

loop of circulating water is itself cooled by evaporation. The circulating water loop must also be resupplied with external makeup water to compensate for evaporation losses.

In the simplest implementation, the water loop is formed by pumping water from a small outdoor pond through heat exchangers to absorb process heat, then discharging the heated water back into the pond from which the evaporation occurs. Controlled external makeup water inflows maintain the pond's water depth. Heat dissipation is then limited to the pond's evaporation rate. A pond's evaporation rate is proportional to its surface area and to an increasing function of its water temperature but is also strongly affected by the air's local pressure, temperature, humidity, and wind speed. High rates of evaporation could in principle be attained if the pond temperature approached water's boiling point, but many heat removal applications cannot tolerate such a high temperature coolant. Since most applications need the water to remain cold, other ways to boost water evaporation rates have been pursued in the technology of cooling towers. As stated in the Cooling Towers chapter of Heat Exchanger Design by Fraas and Osizik, [1965, 241

> "...the heat dissipation capacity per unit of area of a small pond can be increased about twenty times by installing a simple spray system, and about one thousand times by building a cooling tower. Cooling towers have a further advantage over spray ponds in that they reduce the water consumption for a given heat load by a factor of about five, because they can be designed to eliminate the loss of water carried off by the wind in droplet form."

Cooling towers increase evaporation rates by (1) increasing the water's surface area via droplet formation and by (2) enhancing air flow, thus increasing the rate at which water vapor is expelled which in turn decreases average humidity adjacent to the water droplets. A typical cooling tower implementation is depicted in Figure 12-2.

Modern packing material fill is typically fiberglass or polyethylene, but corrugated asbestos and redwood as shown here were formerly preferred. Water drips from internal

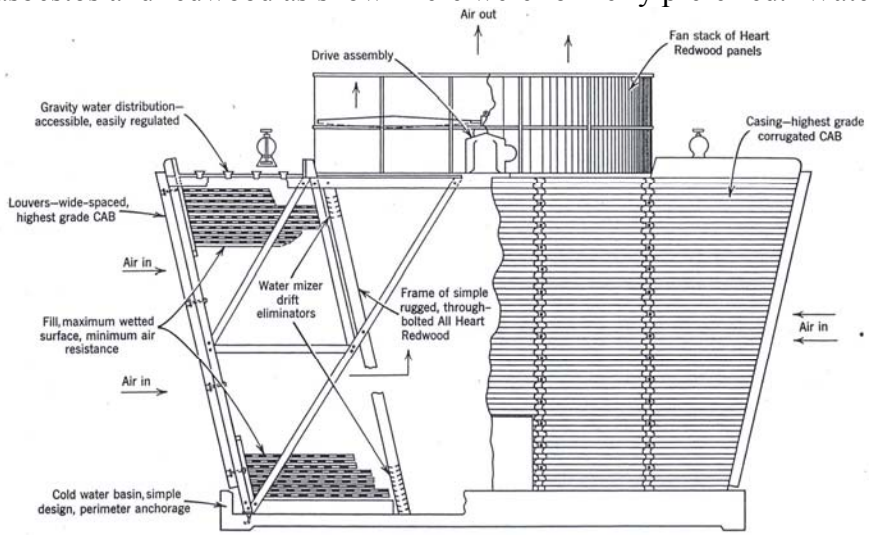


Figure 12-2: Evaporative Cooling Tower (from Fraas and Ozisik)

distribution trays near a cooling tower's top through a descent-slowing internal matrix of *packing material fill* typically arranged in spaced horizontal decks, then the remaining unevaporated water lands in a pond-like lower basin from which it is recirculated to the top. Air enters from the sides through louvers which slant inwards to reduce water droplet loss. Air and water vapor exit through the top.

There are three different design schemes for enhancing the flow of air through cooling towers. Figure 12-2 illustrates the common *induced draft* scheme in which suction from a large fan mounted at the cooling tower's top pulls air through the cooling tower. The *forced draft* scheme in which external blowers located on the tower's sides push air into the tower is only used in very small units since poor performance due to external air recirculation has been historically blamed on the forced draft configuration. The *natural draft* scheme enhances air flow without any fan by constructing a hollow chimney above the cooling tower. Water vapor is lighter than air at the same temperature and is therefore buoyant. Walls of a chimney prevent its lower density enclosed gas from mixing with higher density gas outside, thus producing a difference between the weights of parallel gas columns inside and outside the chimney. The resulting pressure difference produces suction forcing external air inward at the chimney's bottom. Thus, chimneys drive natural airflow provided that the air they enclose is moister or warmer than the air outside. However, to work well a natural draft cooling tower chimney must be quite tall. As a result, the natural draft scheme is only economical for applications where the heat load is large such as typical electric power plant installations. Figure 12-3 shows such natural draft cooling tower chimneys at an electric power station in the U.K.

Not all evaporative cooling schemes are indirect. Steam locomotive engines historically rejected their waste heat by directly exhausting their steam working fluid to the atmosphere at atmospheric pressure, as depicted in Figure 12-4. That open cycle scheme essentially eliminated the need for massive heat rejection equipment. However, it reduced conversion efficiency because the rejection temperature was high (100 C). It also required frequent stops to replenish expendable water supplies which had to be carried in addition to combustible fuel.

Water's evaporative enthalpy change at Mars' low surface air pressure is 10% higher than at Earth's higher pressure, so heat rejection on Mars by water evaporation would in principle be even more effective than on Earth. A cooling pond would work well, and an open cycle steam engine would not sacrifice any efficiency since its rejection temperature could be low. However, the supply of makeup water would be difficult. Mars does not have liquid oceans or lakes and its apparent river beds remain dry. Extremely large amounts of water on Mars have so far been unequivocally found only in solid form as deep polar glacier icecaps. Larger near-surface underground ice deposits as inferred from orbiting spacecraft probably do exist but their use in the quantities needed for evaporative cooling would first require large mining operations. Although water evaporation has been extensively used on Earth, the expected difficulty of obtaining martian water will likely make it too expensive there to routinely expel it into the air for heat rejection purposes alone. Thus, evaporative heat rejection on Mars should not be pursued.

Figure 12-3: Natural Draft Cooling Towers

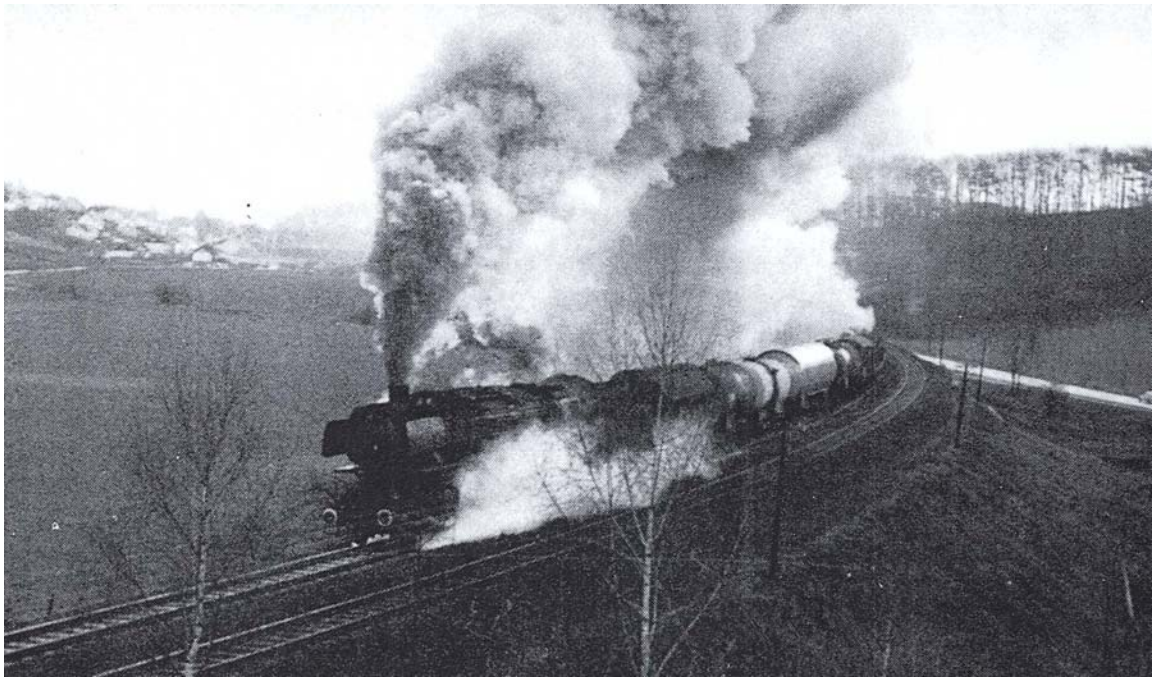


Figure 12-4: Open Cycle Rankine Engine Rejects Waste Heat In Exhaust Steam (Courtesy [Decher])

12.2.1.2 Heat Rejection By Fluid Temperature Rise

Environmental fluids can absorb rejected waste heat without phase change by increasing temperature. A stream of external fluid absorbing thermal power without changing phase while flowing through a heat exchanger exhibits a temperature rise from inlet to outlet. Earth has two convenient environmental fluids, water and air. Mars only has its air.

12.2.1.2.1 Heat Rejection By Liquid Water Temperature Rise

Since Mars does not have any oceans or lakes on its surface, rejecting waste heat into flowing external streams of liquid water is not an option there. It is only briefly considered here for comparison purposes and because it has had such a major role in past and present terrestrial implementations of thermal conversion systems.

Because water at Earth's sea level pressure remains liquid from 0 C to 100 C and because the specific heat of water is 4184 J/kg/K, a heat exchanger rejecting one million watts into a stream of water and using the full 100 C liquid range as its fluid temperature rise would only require a water mass flow rate of (1E6)/(4184)/100 =2.39 kg/s. For smaller temperature rises, Figure 12-5 graphs the water flow rate needed to absorb one megawatt vs. outlet water temperature, for inlet water temperatures typical on Earth.

Liquid water temperature rise is a convenient way to reject heat wherever large amounts of liquid water are found. The scheme has long been used in the condenser heat exchangers of steam engines. It is used today in nuclear powered naval submarines and surface ships and in some nuclear or coal-burning power plants located on seacoasts. As an exercise to estimate the order of magnitude requirements for this type of heat rejection, preliminary design calculations were made for a steam condenser heat exchanger rejecting 10 MW of waste heat into flowing seawater, following design prescriptions given in the book by Fraas and Ozisik. This rating would accommodate the heat rejection needs of a 15 MWth PWR producing 5 MW (6700 hp) output shaft power, which may be enough to propel a very small naval vessel. The resulting condenser layout is shown in Figures 12-6a and 12-6b. Significant findings from this exercise were that the heat exchanger's mass would be about 7000 kg (i.e.7 tonnes) which is 0.7 tonnes/MW heat rejected, and that the "low-head" pumping power required for circulating its seawater would be 7.7 kW, i.e., 0.77kW/MW heat rejected.

12.2.1.2.2 Heat Rejection By Earth-Air Temperature Rise

The specific heat of Earth's dry air is about 1000 J/kg/Kelvin, so the mass flow rate needed for air cooling is about four times the mass flow rate needed for water cooling with identical fluid temperature rises. More precisely, the enthalpy content of Earth's dry air is plotted versus temperature in the Figure 12-7 in which the zero point is taken to be Earth's average surface temperature of 288 Kelvins (about 15 degrees Celsius).

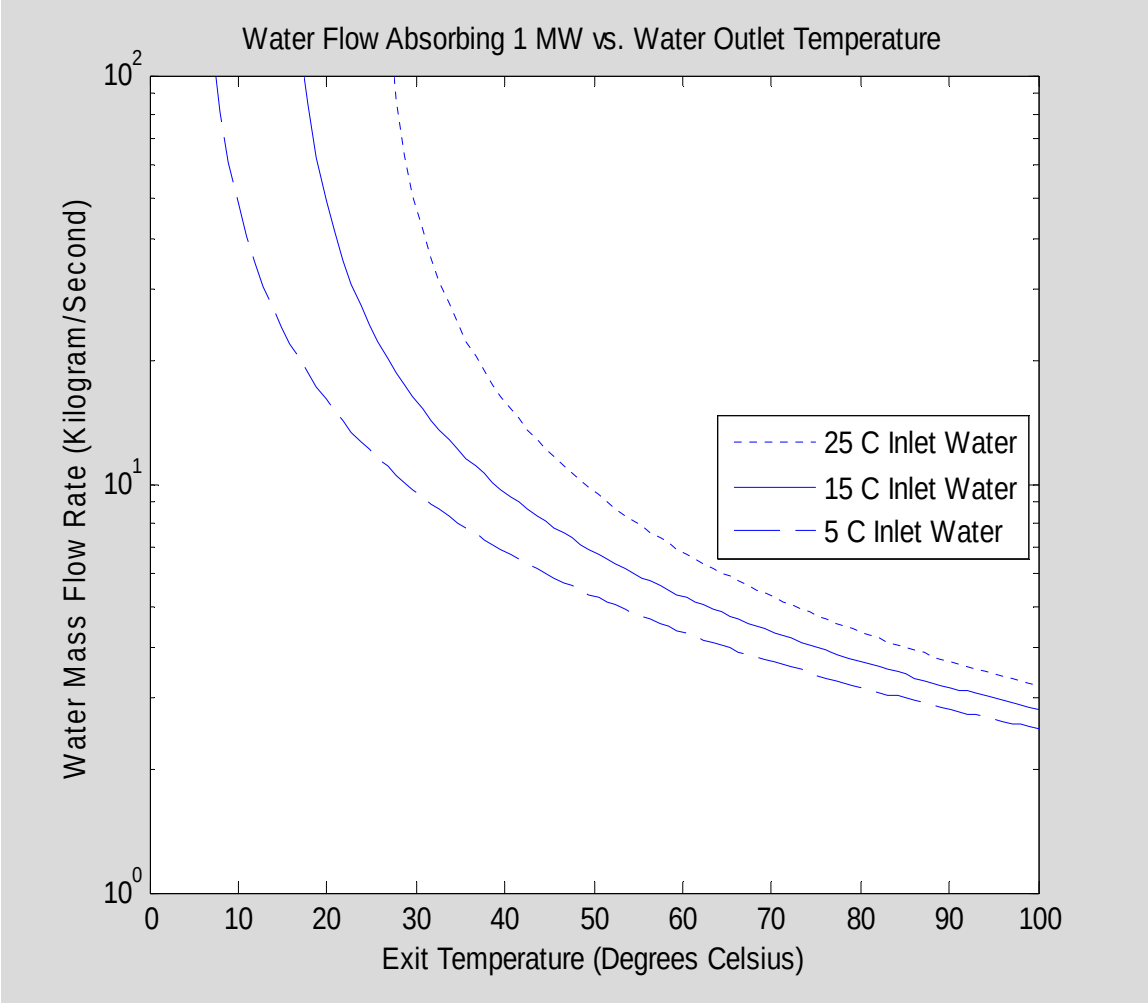
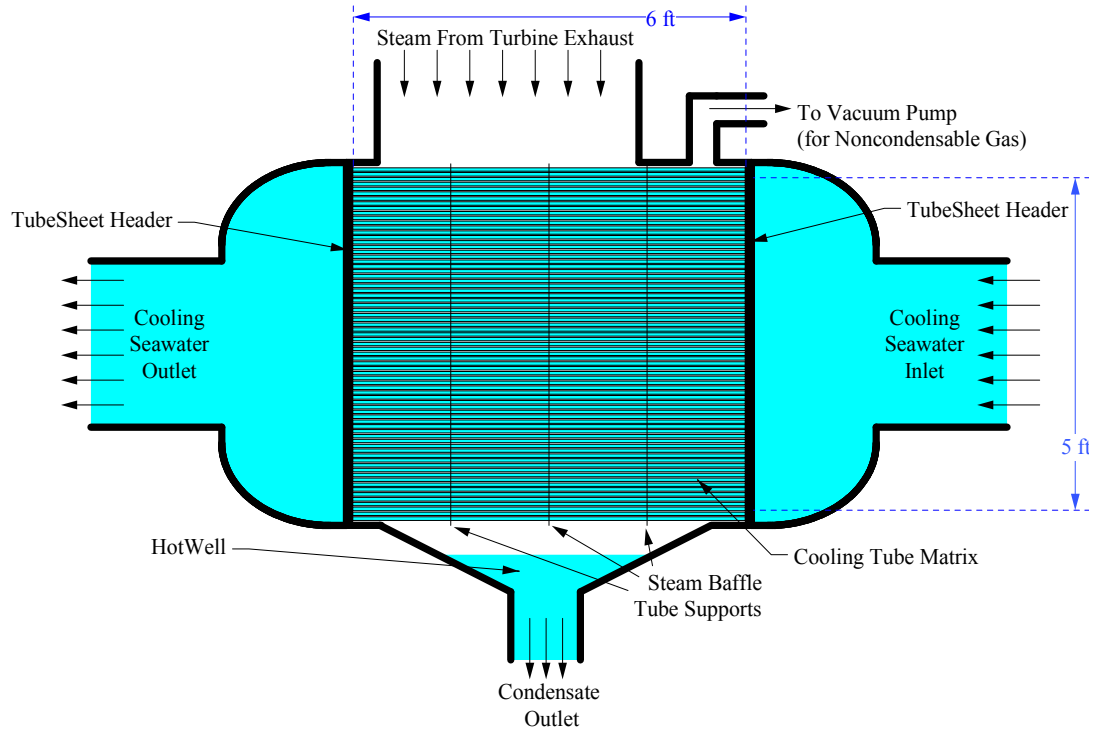
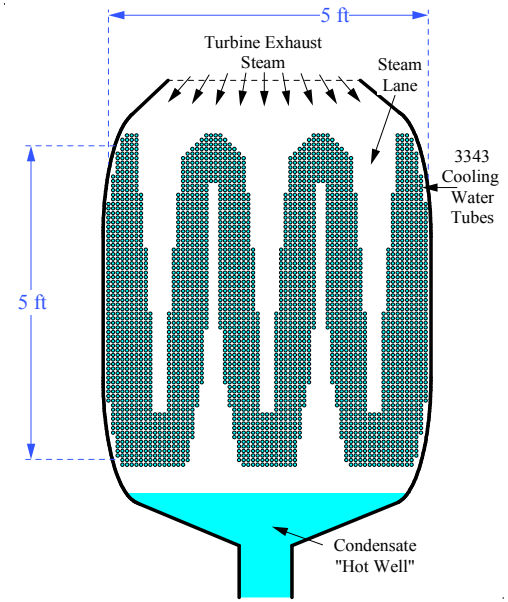


Figure 12-5: Water Mass Flowrate Vs. Outlet Temperature To Absorb 1 MW Heat



(a) Side Cross Section



(b) End Cross Section

Figure 12-6: Cross Sections of 10 Megawatt Steam Condenser Example

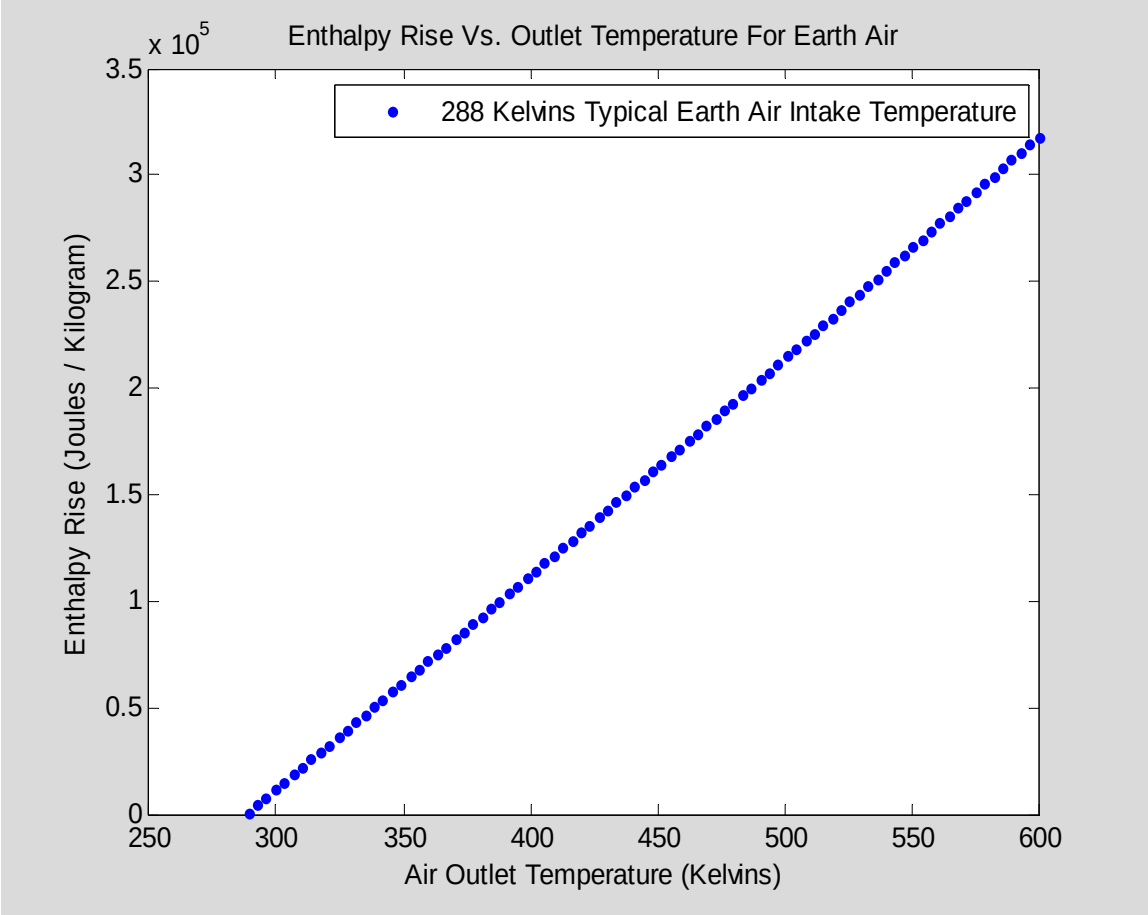


Figure 12-7: Earth Air Enthalpy Rise Versus Temperature

The reciprocal of this enthalpy information is the mass flow rate needed per unit of heat absorbed. In the Figure 12-8, the Earth air intake mass flow rates needed to absorb 1 megawatt of heat are plotted vs the resulting air exhaust temperature.

Thus, about 80 kg/sec dry airflow can absorb one megawatt with only a 12 C air temperature rise, 10 kg/sec dry airflow can absorb one megawatt of heat with a 100 C air temperature rise, or 5 kg/sec dry airflow can absorb one megawatt of heat with a 200 C air temperature rise. At Earth's sea level pressure and 15 C average temperature, the dry air density is 1.225 kg/m^3. Thus, volumetric air flow rates per megawatt of heat transferred are about 65 m^3/sec for a 12 C air temperature rise, 8.2 m^3/sec for a 100 C air temperature rise, or 4.1 m^3/sec for a 200 C temperature rise. Thus, higher air temperature rises allow reduced airflow and may permit smaller air cooler heat exchangers, but temperatures must also be limited consistent with thermal requirements of the cooling application.

Since hot air expands and rises as it is heated, natural convection helps prevent heated gas from reentering the air cooler heat exchanger. In principle, natural convection alone could also force an air stream through a heat exchanger. However, natural convection air cooler heat exchangers are almost never economical and so are rarely if ever seen. Instead, the common design practice for air cooler heat exchangers is to augment natural convection with powered external fan blowers in order to drive a much higher air flow rate than natural convection alone could provide. A relatively small amount of fan power enormously increases heat transfer effectiveness.

Actual air cooler heat exchangers are classified as either forced draft or induced draft systems depending on whether the fan blows air into the heat transfer media or sucks air from the media. In either case, the media is normally a multilayer array of parallel finned tubes carrying the flowing fluids being cooled. In either case, the direction of air flow through the media usually has a strong upwards vertical component (so that thermal buoyancy effects are helpful). Thus, forced draft air coolers have fans located below while induced draft air coolers have fans located above. Figures 12-9 and 12-10 (taken from the book by Frass & Orsizik) depict each of the two types.

Recent texts on heat exchanger technology note that over the past 50 years, the use of air cooler heat exchange technology for heat rejection has replaced the older evaporative water cooling systems at many industrial sites globally. The change has been almost complete in some fields, for example in petrochemical refineries. It has also become the standard technology for smaller scale consumer applications of heat rejection, such as in the heat rejection components of home air conditioning systems

Heat exchangers operating with air at atmospheric pressure are not the only way to reject waste heat into the air. **A much more common way to reject heat into the air is found in open-cycle thermal conversion systems such as automobile or aircraft engines.** In open-cycle engines, a stream of air is drawn into the engine's air intake while hotter gas exits via the exhaust. **No separate heat exchanger is needed** for the thermal conversion

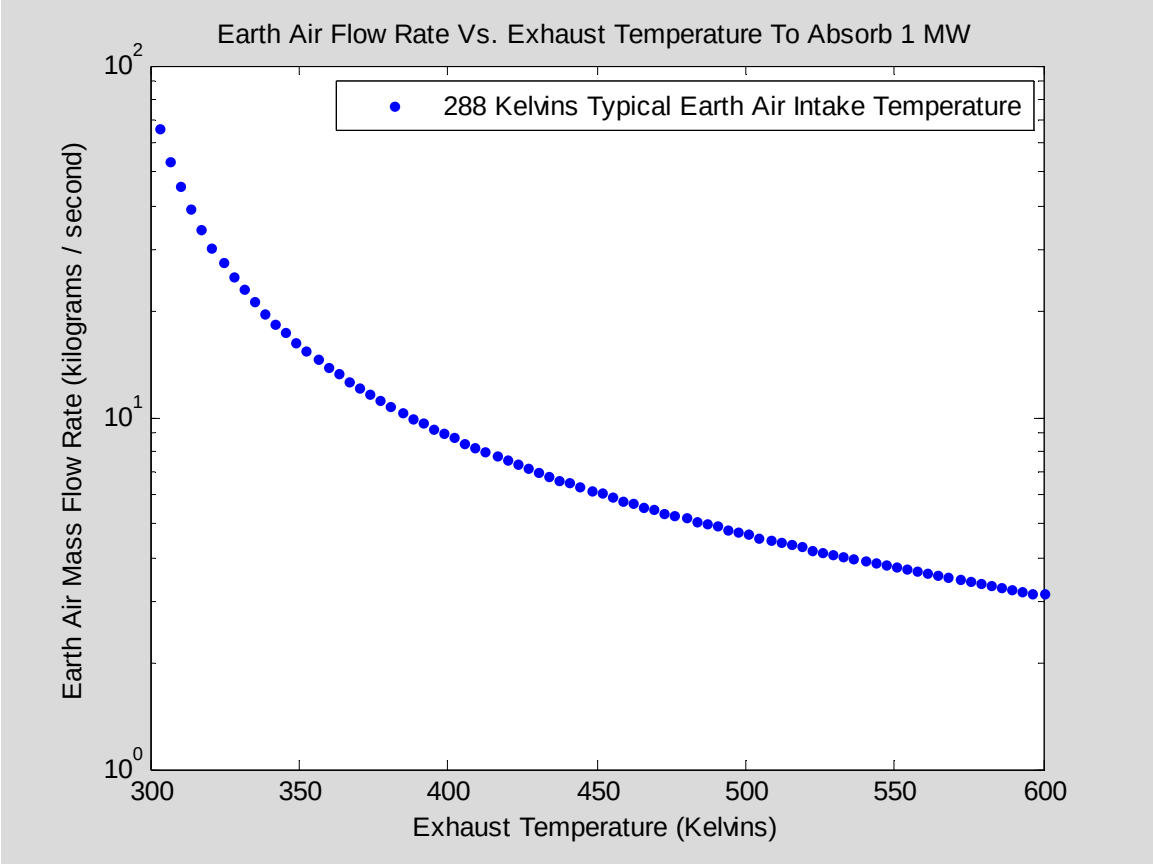


Figure 12-8: Mass Flow Rate of Earth Air To Absorb 1 MW, Vs. Exhaust Temperature

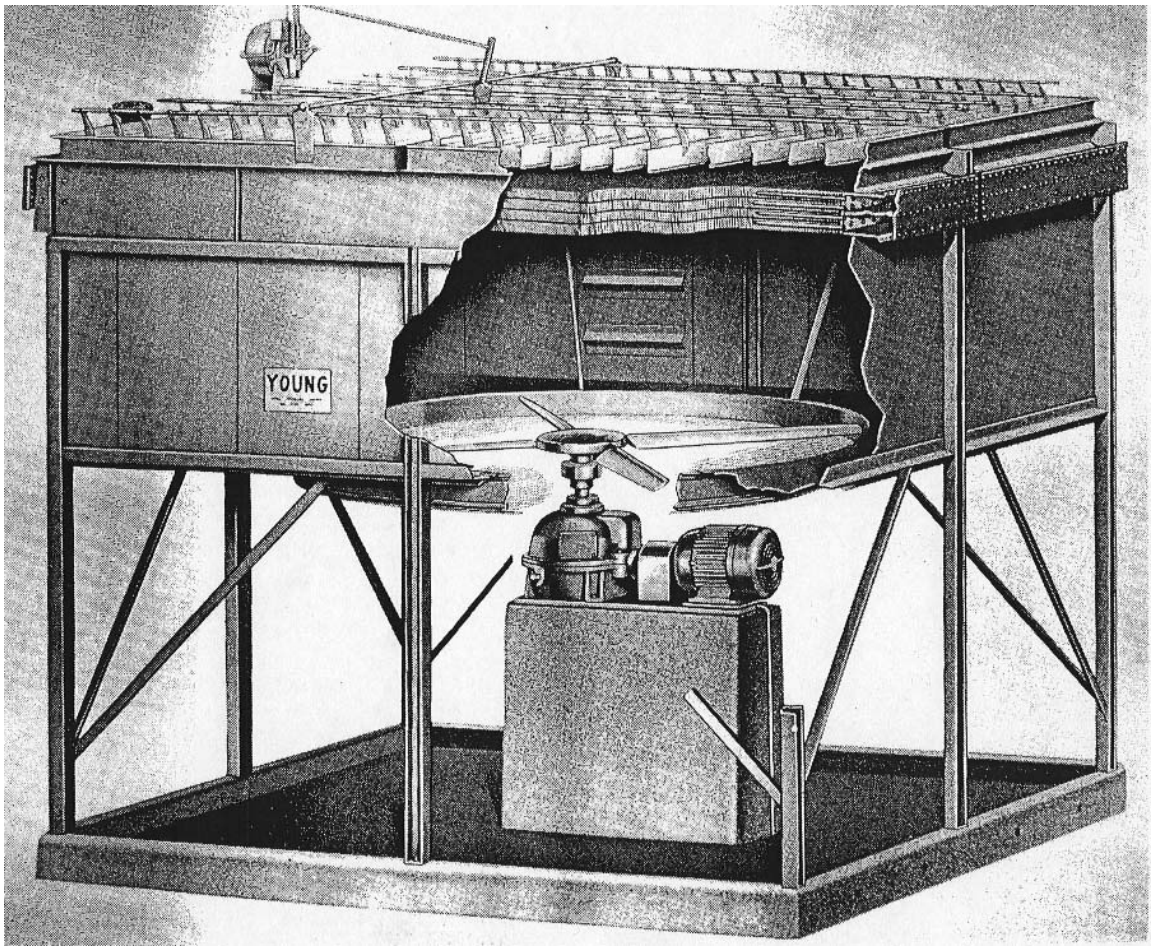


Figure 12-9: An Industrial Forced-Draft Air Cooler Heat Exchanger

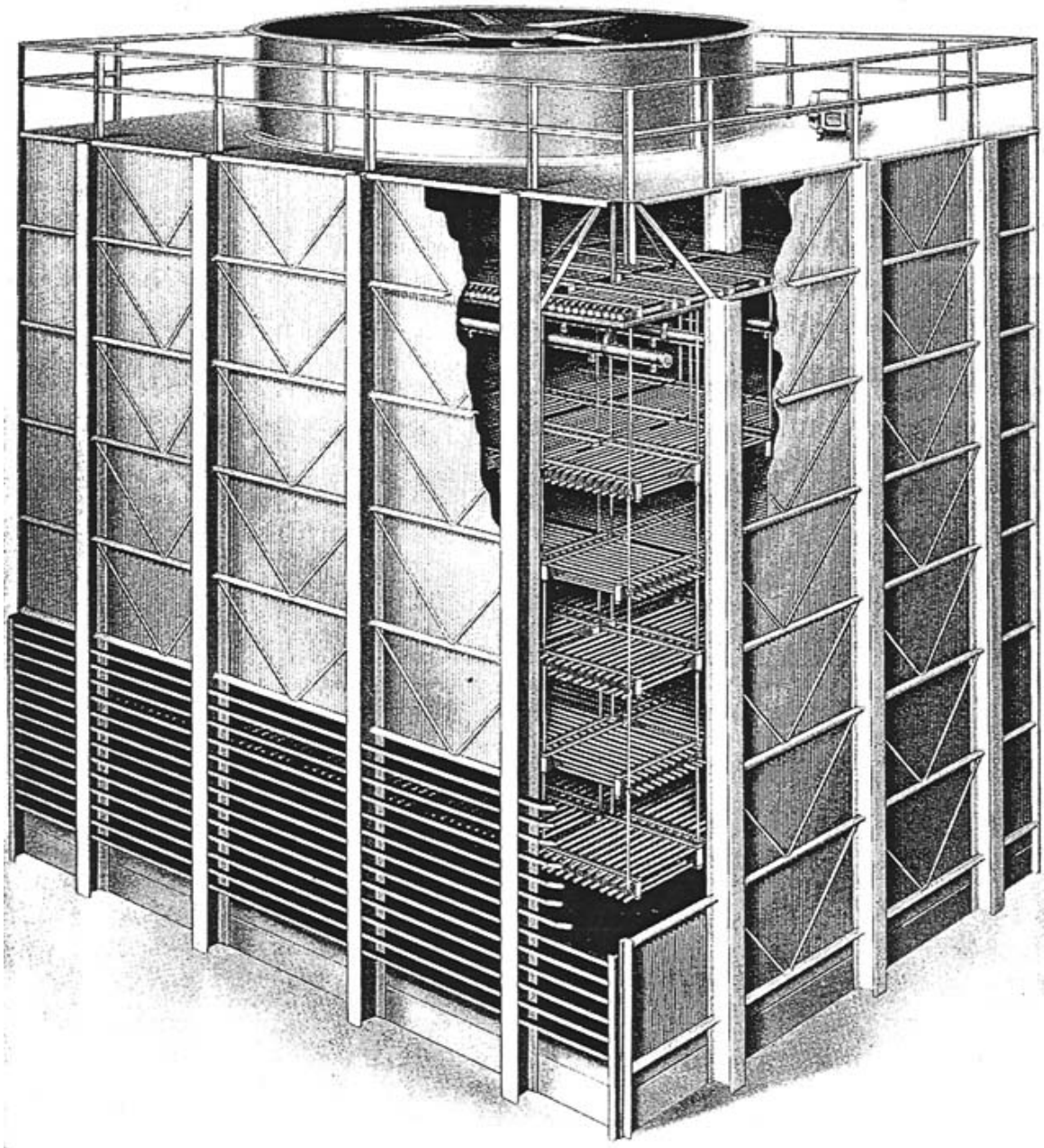


Figure 12-10: An Industrial Induced-Draft Air Cooler Heat Exchanger

waste heat. Instead, the air is heated as part of the thermal conversion scheme, and the waste heat is rejected simply as the enthalpy difference of inlet and outlet gas streams. In the case of open-cycle engines powered by combustion, the situation is only slightly more complicated. A proper accounting of incoming enthalpy must also include the fuel while the outgoing enthalpy must also include the combustion products. Much of the enthalpy difference between inlet and outlet streams develops because the gas exhaust is hotter than the air intake, but the combustion products also contribute to the waste heat rejection without further increasing the exhaust temperature.

12.2.1.2.3 Heat Rejection By Martian Air Temperature Rise

This provides a practical way to reject waste heat on Mars' surface.

Martian surface temperatures are colder than Earth temperatures and also more variable. Mars' average surface temperature has been quoted as being 210 Kelvins or 218 Kelvins. Infrared radiometer surface temperature measurements from orbit show daily maximum ground soil temperatures on the equator are typically about 300 Kelvins (=27 C or 80 F) and are slightly warmer in certain locations. Typical diurnal temperature variations are 50 to 60 Kelvins between night and day, and there are much larger seasonal temperature variations at locations away from the equator. Radiometer data show minimum surface temperatures approach 130 Kelvins at Mars' poles late in the long winter season of continuous darkness.

However, the range of *air* temperature variations near Mars' surface is likely smaller than the range of surface temperatures, because solar heating and radiative cooling both couple directly to the ground more than they couple to air, because the buoyancy of surface-heated air causes it to immediately interchange with colder air above it, and because the carbon dioxide forming most of Mars' atmosphere freezes out and precipitates as solid dry ice frost at pressure-dependent temperatures in the 140 Kelvins to 150 Kelvins range. For these reasons, 215 Kelvins is assumed for the present design work to be the average martian near-surface air temperature while the martian air temperature's full range assumed herein for design purposes extends from 150 Kelvins to 280 Kelvins.

Figure 12-11 graphs shows enthalpy rise versus air exhaust temperature for this full expected range of martian air intake temperatures and also compares it with the enthalpy rise of Earth air at its typical intake temperature. It shows that typical martian air can absorb more heat per unit mass than typical Earth air if heated to the same exhaust temperature, mainly because Mars' intake air temperature is colder.

These enthalpy rise curves lead to Figure 12-12, an associated graph showing the mass flow rate of martian air required to absorb one million watts, plotted against the discharge temperature of the heated air exhaust. Figure 12-12 includes a corresponding Earth air graph for comparison.

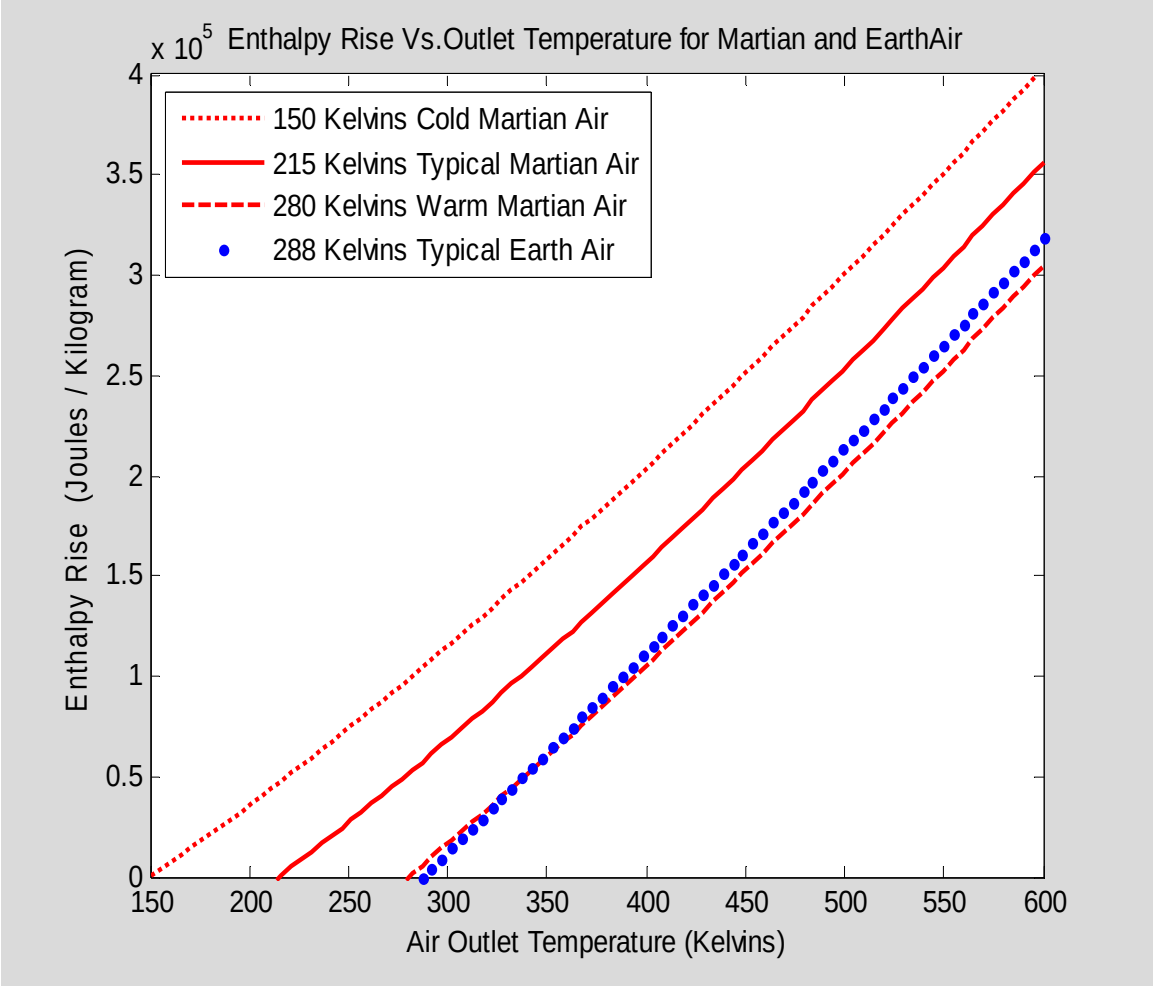


Figure 12-11: Mars & Earth Air Enthalpy Rises Vs Exhaust Temperature

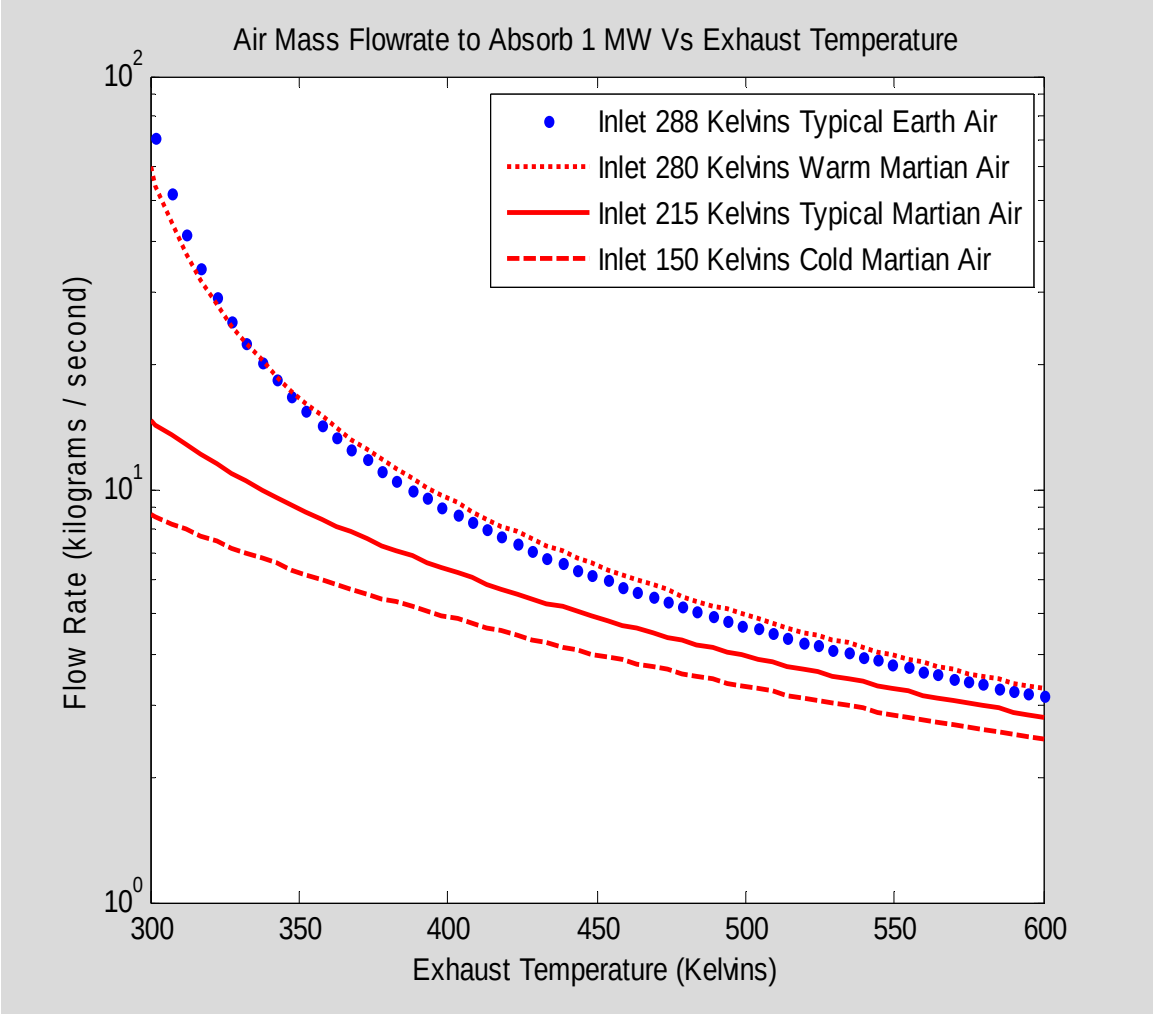


Figure 12-12: Air Mass Flow Rates To Absorb 1 MW Heat, Mars Vs Earth

These plots show that heat rejection by martian air temperature rise could be just as effective as heat rejection to terrestrial air, and may be even better since the martian air is colder. Similar to terrestrial designs. e.g., Figures 12-9 and 12-10, well-designed air cooler heat exchangers for Mars would employ powered fans to move the martian air through their media, which would likely also be configured as 3D arrays of finned pipes running between parallel headers. In contrast to terrestrial designs, fan-powered air speeds for martian designs would be higher; the media arrays would be physically larger with taller fins and with wider inter-fin spacings for similar rates of waste heat rejection and similar levels of fan power consumption.

12.2.2 Heat Rejection By A True Radiator

A true radiator would be less effective than rejecting heat to the martian air. For many locations beyond Earth, the only viable long-term option for waste heat rejection is to use a true radiator. In deep space locations the only alternative way to reject heat would be to discharge a mass of fluid to carry the heat away, but this could only be done temporarily before the fluid would be gone. The main difficulty with a true radiator is that either a large radiator size or a high radiator temperature is needed to reject the large amounts of waste heat which must be rejected for typical engines.

Figure 12-13 graphs the maximum possible radiative power density vs radiator temperature. Alternatively this same relationship determines the theoretical minimum radiator area required to reject one megawatt of heat, which is graphed in Figure 12-14. .These plots show that radiator heat flux varies enormously with radiator temperature, and that large radiator areas would be needed for the amounts of power typical of manned vehicles on Earth. It should be noted that the type of area needed for a radiator is somewhat special. It cannot be subdivided into subareas such as arrays of parallel plates, since any portions of a radiator surface which face each other are counterproductive. To be effective, the entire radiator surface must face outward.

However, the efficiency of the thermal conversion cycle also varies enormously in the opposite sense, dropping to low values for high waste heat rejection temperatures. To examine the resulting tradeoff we initially ignore the temperature drops required to transfer heat from the thermal conversion process gas to the radiator's surface as well as the temperature change of the gas itself. Requiring instead that the radiator dispose of the engine's waste heat flow at an idealized Carnot thermal conversion system's single cold exhaust temperature, the above idealized equations can be algebraically combined to solve for the engine output power per unit of radiator area:

$$\frac{P}{A} = \left(\frac{\eta_e \varepsilon \sigma \left(T_H - T_C\right)\left(T_C^4 - T_{\text{Environment}}^4\right)}{T_H - \eta_e \left(T_H - T_C\right)} \right) \qquad (12\text{-}7)$$

To numerically evaluate Equation 12-7, we take the high temperature heat source temperature to be 1300 Kelvins since this temperature was adopted in the past as the design temperature for space nuclear power projects such as SP100. We assume the.

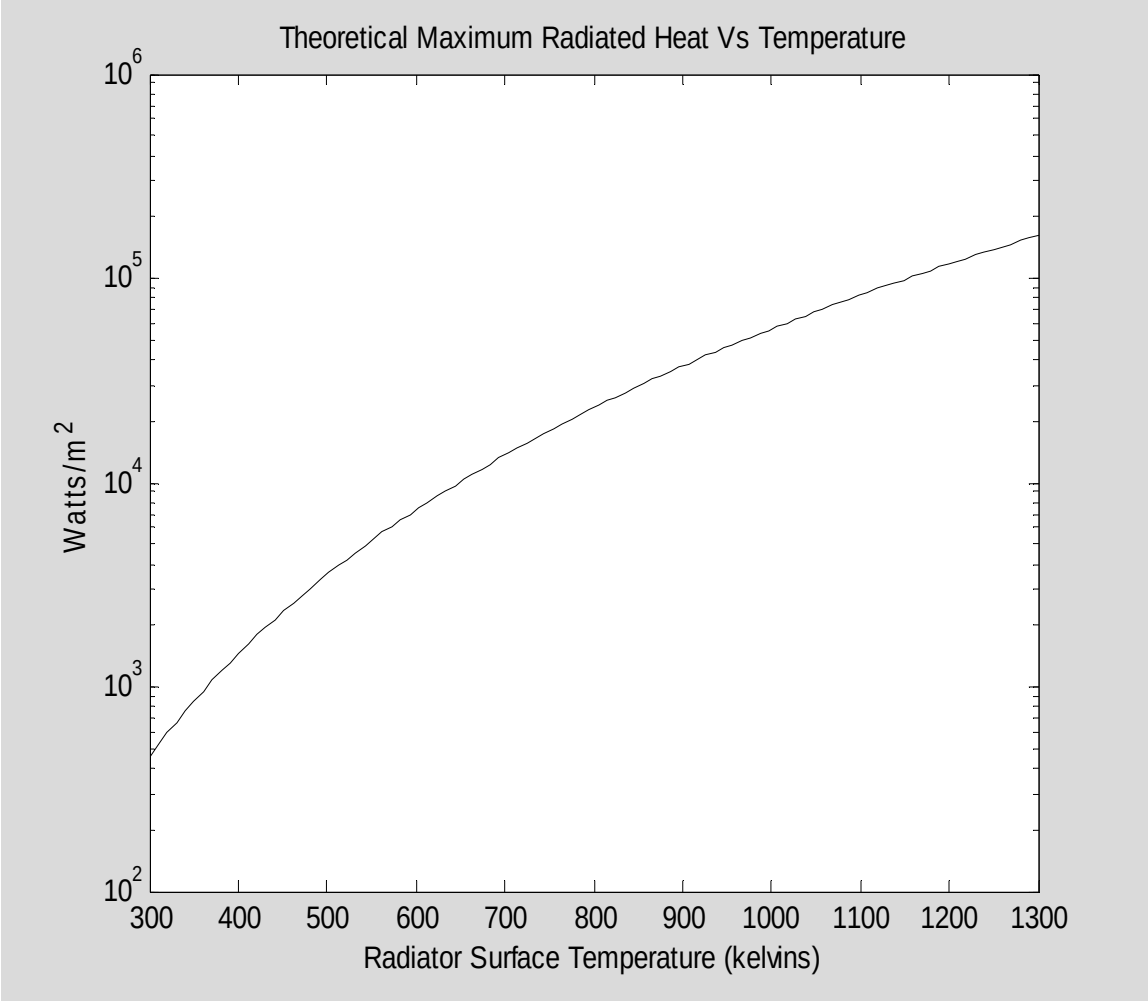


Figure 12-13: Radiative Power Density Vs Temperature For Ideal True Radiator

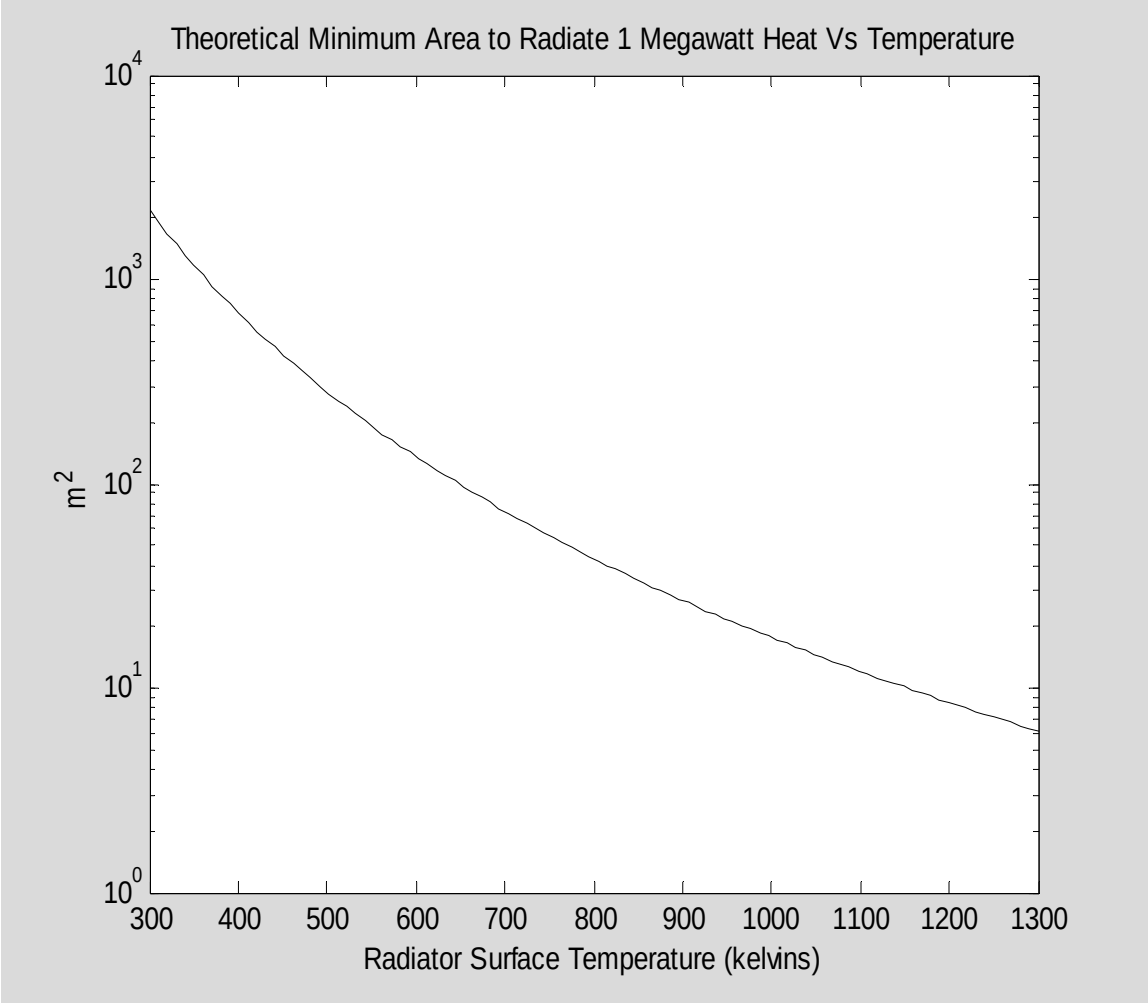


Figure 12-14: Necessary Radiator Area Per Megawatt of Heat Rejected Vs Temperature For Ideal True Radiator

effective radiative temperature of the environment is 200 Kelvins, which may be approximately correct on Mars We take the radiator surface emissivity to be ε=0.55, a typical value tabulate for stainless steels in the temperature range from 500 through 1300 Kelvins [Modest 1993, 762-779]. Figure 12-15 plots the resulting calculated engine power per unit radiator area versus radiator temperature for several assumed isentropic thermal conversion system efficiencies.

The Figure 12-15 plots show that under the idealized parameters assumed, 4 to 6 horsepower per square meter of radiator area might be obtained by optimizing the radiator's design temperature, and perhaps as much as 8 horsepower per square meter might be obtained by also enhancing isentropic efficiency of the conversion system to 70%, an unusually high level. That would raise the actual thermal conversion efficiency to about 15%. Now, mobile engine efficiencies in the 10% to 15% range are low by terrestrial standards. Indeed, the optimum radiator temperature for exhausting waste heat of about 1000 Kelvins as shown in the above plots is more than hot enough to drive a terrestrial energy conversion system. However, if there were no alternative, e.g., as for a mobile vehicle on Luna, Earth's airless moon, the capabilities provided by such an energy conversion systems using radiators to dispose of the waste heat would be useful.

On the other hand, the available engine power would drop substantially if it were necessary to reduce the reactor's design temperature, as the Figure 12-16 illustrates.

As shown in Figure 12-16, at a reactor heat source temperature of 1000 Kelvins only about 1.5 to 2.0 horsepower of engine power per square meter of radiator could be expected, with perhaps up to 2.8 horsepower per square meter if the assumed Carnot style of conversion system's isentropic efficiency were enhanced to 70%. Thus, dropping the designed reactor temperature from 1300 to 1000 Kelvins would reduce vehicle engine power by a factor of about three, if a radiator is used to dispose of the waste heat. This rapid fall-off of engine power with reactor temperature arises because radiation increases as the fourth power of radiator temperature. Other heat rejection schemes relying on convective heat transfer to air or water do not have this nonlinear temperature dependence. Thus, for radiative disposal of waste heat to be attractive, the nuclear reactor should be designed to provide heat at a very high temperature, e.g., 1300 Kelvins or more. Even then, the attractiveness would disappear if a better option were available.

To function effectively a radiator's surface should be exposed directly to the sky. At its optimized design temperature near 1000 Kelvins the radiator, visibly glowing red, would be hazardous for a suited astronaut to touch. For these reasons it seems appropriate for mobile surface vehicle engine applications to elevate the radiator above where astronauts might accidentally touch it. The radiator should be mounted to extend horizontally above the mobile vehicle it powers, i.e., above the reactor, its thermal conversion system and the roof of the manned compartment. In this location the radiator's surface area would match the footprint planform area of the entire vehicle. Below the radiator there would need to be protective layers of thermal insulation or simply radiative shields.

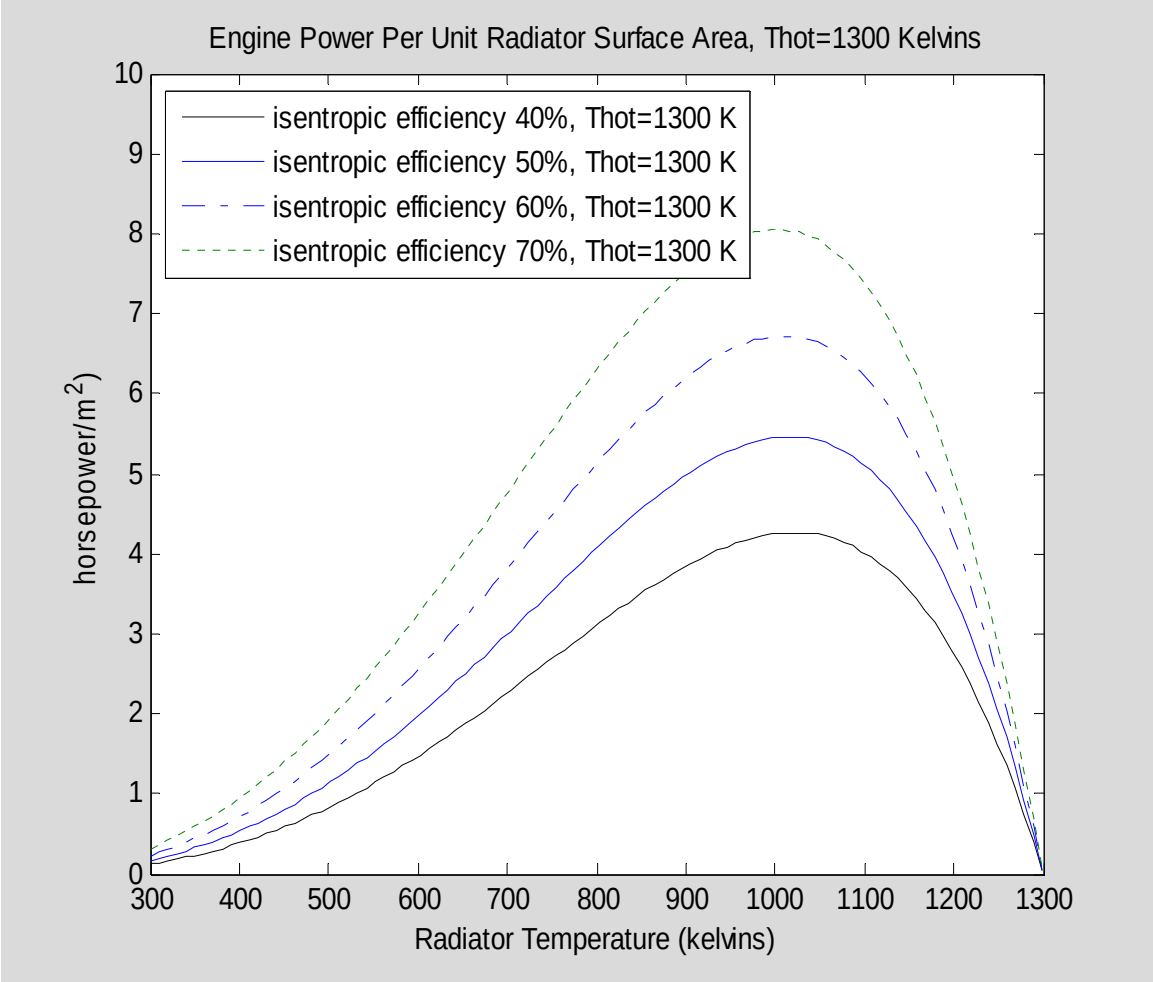


Figure 12-15: Theoretically Achievable Engine Power Per Unit Surface Area Of Radiator, Thot=1300 Kelvins

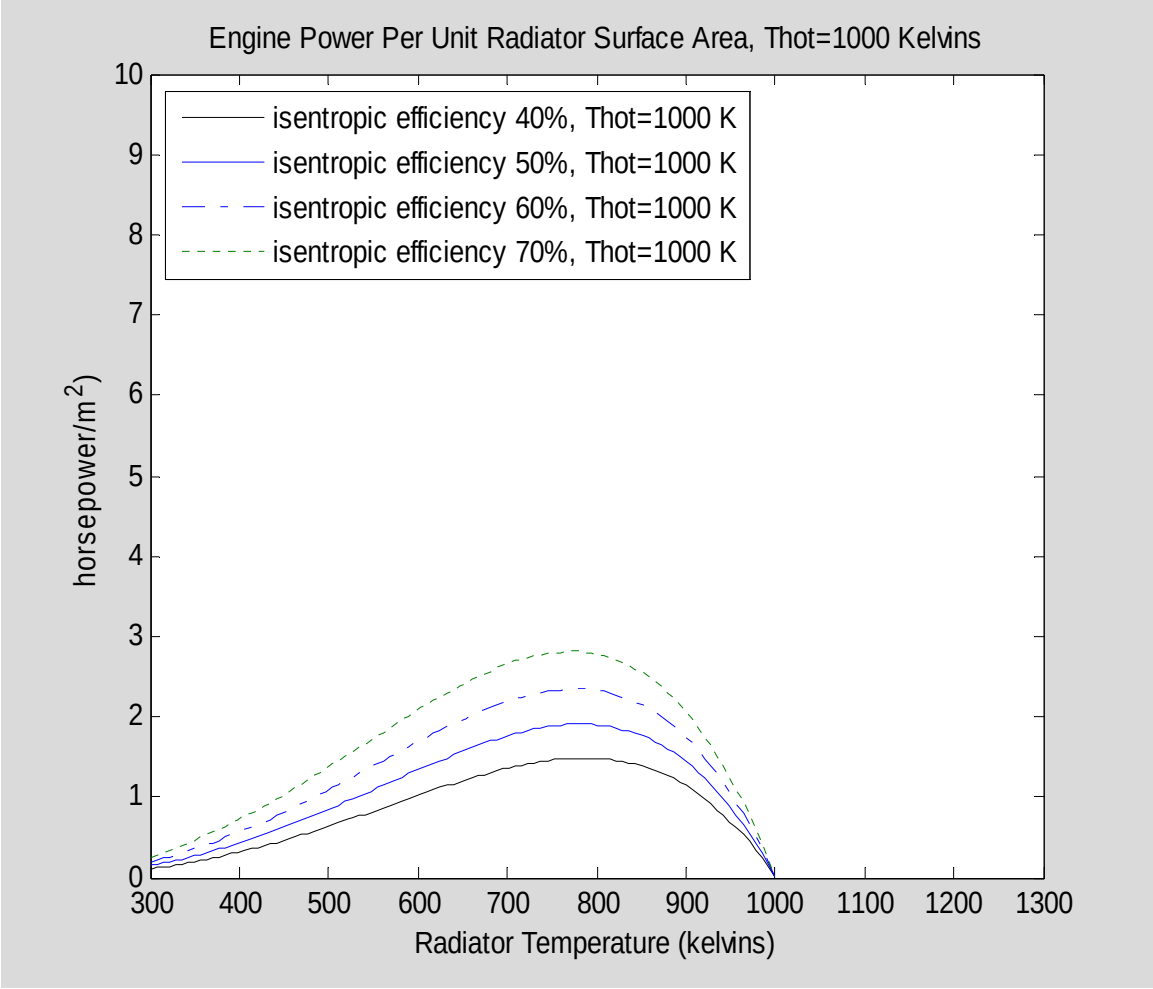


Figure 12-16: Theoretically Achievable Engine Power Per Unit Surface Area Of Radiator, Thot=1000 Kelvins

Typical automobiles have planform areas in the range from 5 to 10 square meters. If mobile vehicles for Luna with heat exhaust radiators covering their tops were of similar size, their maximum engine powers operating from 1300 Kelvin heat sources would be in the range from 20 horsepower to 80 horsepower. Such power levels are lower than typical engine power ratings for most modern automobiles and seem far too limiting for more demanding off-road applications in a hostile extraterrestrial environment. On the other hand, a car-sized vehicle would be too small for long distance excursions on either Luna or Mars. A larger vehicle, perhaps shaped as a rectangular solid, 4 meters wide and 16 meters long including the nuclear reactor and thermal conversion system, would be large enough to house two or three astronauts and their recycling life support systems during long distance excursions. Its planform area would be 64 square meters and a radiator mounted above it would have the same surface area. According to the Figure 12-15 plots, this is enough radiator area to dispose of the waste heat from an engine in the 250 to 380 horsepower range, or as much as 500 horsepower if the engine's isentropic efficiency were enhanced to 70%. Such a vehicle would be useful.

On Earth, large equipment used by mining and road building industries for excavation, e.g., bulldozers, frequently have engines developing thousands or even tens of thousands of horsepower. Although such high powered mobile engines would be useful on either Luna or Mars, it is not practical to dispose of their large waste heat through radiators small enough to fit above the vehicles. Engines developing significantly higher levels of power per unit area thus cannot rely on a true radiator to dispose of their waste heat.

Unfortunately, the above conclusions about radiative heat rejection may be optimistic in their estimates of radiative heat rejection performance because they ignore two important types of temperature drop which will unavoidably be present. These are (1) the changing temperature of the thermal conversion process working fluid as it passes its waste heat to the radiator and (2) the steady temperature difference due to heat transfer temperature drops between the working fluid and the radiator's external surface. These are discussed further below.

It should also be mentioned that although several thermal conversion system designs powered by nuclear heat sources have flown in unmanned deep space missions, most have developed conversion efficiencies in the 2% to 5% range. None has ever reached even 10% efficiency.

In reality, ideal Carnot systems are not feasible since there is no simple design of a thermal conversion system which isothermally receives high temperature heat and isothermally rejects colder waste heat. The performance of such ideal systems can only be approximately approached by either greatly oversizing energy conversion components while using a high mass flow rate of the thermal conversion process working fluid or by combining multiple thermal conversion system stages with appropriate interfacing heat exchangers. Unfortunately, all such approaches increase mass.

An actual engine using a nuclear reactor heat source and radiative disposal of waste heat would employ an encapsulated practical closed-cycle energy conversion scheme such as a version of the Brayton cycle, using a compressed gas such as helium for its working fluid. Heat rejection would be accomplished by routing the gas on the low pressure side through heat exchanging tubes which are thermally well-connected to the radiator. Just as high temperature heat is added to the gas by raising its temperature, so colder waste heat is rejected from the gas by reducing its temperature. Thus, the average temperature of gas in the radiator's attached gas tubes may be considerably colder than the temperature previously assumed in Eq. 12-7. Also, for heat to flow out of the gas into a wall surface, then through the metal wall to an external radiating surface, there must be an additional temperature drop proportional to the heat flux. Thus, the average temperature of the radiating surface would be colder than the average gas temperature inside the radiator's tubes.

To summarize the situation for radiative heat rejection, it is worth pursuing if no better alternative is available. It appears to be the only option for nuclear powered lunar vehicles. Air-breathing engines may be better for martian vehicles.

CHAPTER 13
13 OPEN BRAYTON CYCLE (OBC) USING MARTIAN AIR

13.1 Summary

The Open Brayton Cycle is analyzed in this chapter using the actual nonlinear thermodynamic properties of the martian air, first assuming ideal components, then considering practical inefficient components.

13.2 The Martian Air Model

The composition of martian air as determined by the Viking Mars Lander spacecraft is as given in Table 2-3, which lists 10 different chemical components. For the present calculations, martian air is simulated by a simplified gas mixture herein called MarsMix, which contains only three components and ignores the other seven trace gases. The three included gases are carbon dioxide (95.7%), nitrogen (2.7%) and argon (1.6%). Since these are readily available and relatively inexpensive gases, it would be feasible to prepare some of this martian air simulant in order to perform engineering experiments in a laboratory. However, that has not been done as part of this thesis work. For the present purpose the reduction from 10 to 3 ingredient gases slightly simplifies the tasks needed to estimate martian air properties from tabulated pure chemical gas properties. The property estimation has been accomplished as documented and fully explained in Appendix B. The resulting MarsMix gas properties have been implemented as MATLAB m-file subroutines which have been used in this chapter's calculations.

Based on its composition, the mean molal mass of MarsMix is 0.0435131 kg/mole. Its density obeys the ideal gas law,

$$\rho = \frac{p}{RT} \tag{13-1}$$

where the gas constant for MarsMix is R=191.06 J kg^{-1} Kelvin^{-1} and where ρ, p, T represent density, pressure, and absolute temperature, respectively.

As discussed in Chapter 2, ambient air temperature and pressure conditions at Mars' surface vary considerably with local surface elevation, with the martian season, with the time of day, with latitude and perhaps even with longitude. At typical martian air pressures the carbon dioxide which forms the main constituent of martian air precipitates out as solid dry-ice frost at temperatures between 140 and 150 Kelvins, so air temperatures colder than 140 Kelvins are not possible. Based on satellite-based infrared measurements of the ground temperature, it is assumed herein that martian near-surface air has a global average temperature of 215 Kelvins, but that at some times and locations the air may get as cold as 150 Kelvins while at other times and other locations it may get

as warm as 280 Kelvins. Thus, near-surface air which will be drawn into the Brayton Cycle engine is assumed to always have a temperature in the range, 215±65 Kelvins.

As described in Chapter 2, daily average pressure at the two Viking Lander sites observed over several martian years ranged from 690 to 1010 Pascals, but these were both near the reference altitude analogous to Earth's "sea level". Based on the 11.1 km estimated e-folding height of the martian atmosphere, surface locations within the ±7 km altitude range which includes most of Mars surface except its taller mountains would have pressures between 0.52 to 1.9 times the reference altitude values. Thus, it is assumed that near-surface air drawn into the Brayton Cycle engine while traveling globally on Mars surface will have a pressure in the range from 350 Pa to 2000 Pa.

For this range of martian ambient air temperatures and pressures, Eq. 13-1 yields the Figure 13-1 contour plot of calculated martain air densities..

Although engine air intake conditions corresponding to any point on the Figure 13-1 plot will be assumed possible, points in the upper left or lower right may be unlikely. Low altitude locations are rarely cold and high altitude locations are rarely warm. .

Among the several traditionally considered thermodynamic properties, the one most closely related to those of Figure 13-1 is the specific volume, V. It is simply the reciprocal of the gas density, i.e., the volume per unit mass of the gas. Figure 13-2 graphs its calculated values for the MarsMix martian air simulant for the same range of ambient air conditions expected for most of Mars' surface.

It is necessary to model MarsMix gas properties over a much larger range of temperature and pressure conditions in order to estimate thermodynamic processes taking place within the nuclear powered engine. Although the density and specific volume properties of Figures 13-1 and Figure 13-2 obey the Equation 13-1 ideal gas law over that larger range, other thermodynamic properties do not follow such a simple rule and must be analyzed using numerically tabulated functions. As discussed in Chapter 4, the molten lithium to be used as a heat transfer liquid is only compatible with the refractory metals chosen to contain it for temperatures up to 1643 Kelvins (i.e., 1370 C), and is not chemically compatible for long duration exposures at a somewhat higher temperature. Therefore, the engine design will not require the molten lithium's temperature to ever exceed 1643 Kelvins. In turn, the martian air, which will be heated to its highest temperature by heat conducting through a solid heat exchanger wall from the molten lithium, will never exceed or even reach a temperature of 1643 Kelvins. Therefore, 1643 Kelvins is chosen here as an adequate upper limit for calculations of the Brayton Cycle using MarsMix gas. A lower limit of 150 Kelvins is chosen as the coldest martian air expected to ever be encountered. Thermodynamic properties of MarsMix are modeled as tabulated functions which include the range from 150 to 1643 Kelvins (See Appendix C).

For a perfect ideal gas the specific heat at constant pressure, c_p, and the specific heat at constant specific volume, c_v, are each functions of temperature only, i.e., they are each

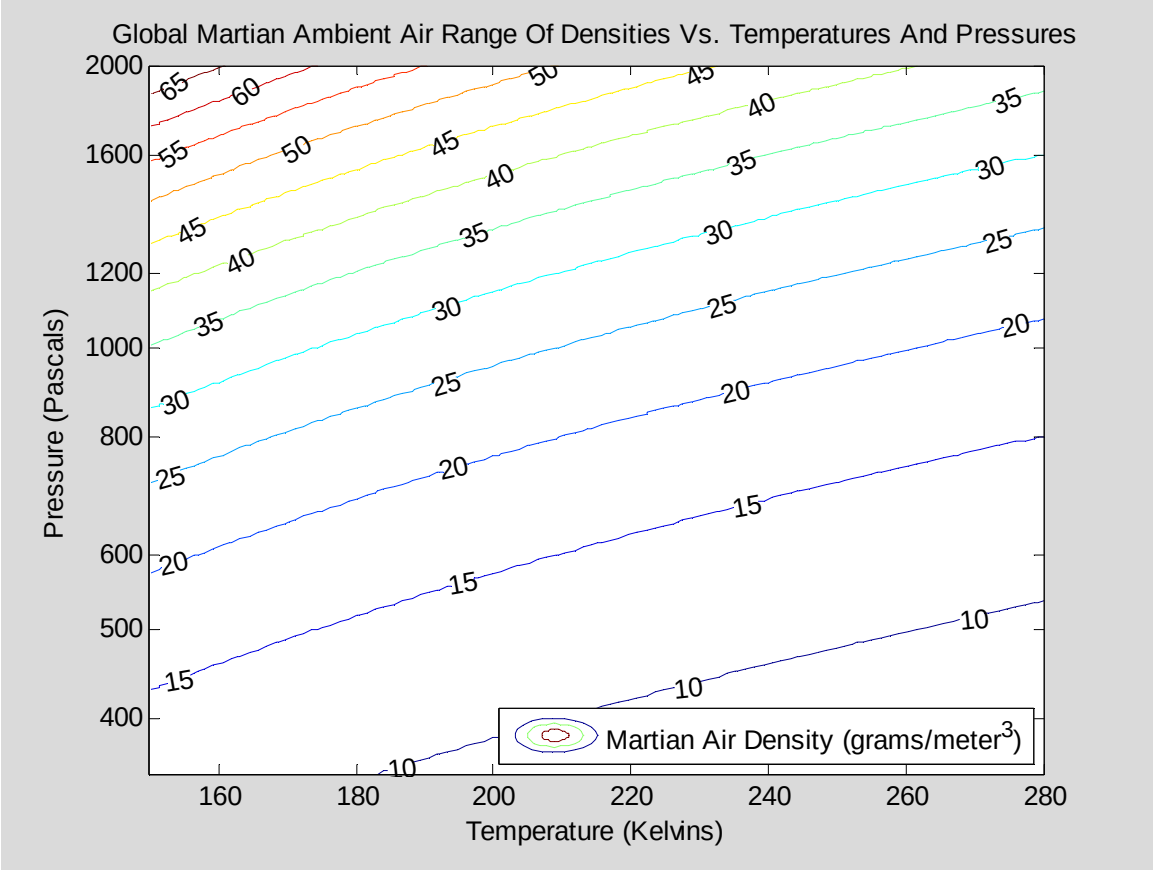


Figure 13-1: Global Near-Surface Ambient Martian Air Density, Pressure, and Temperature

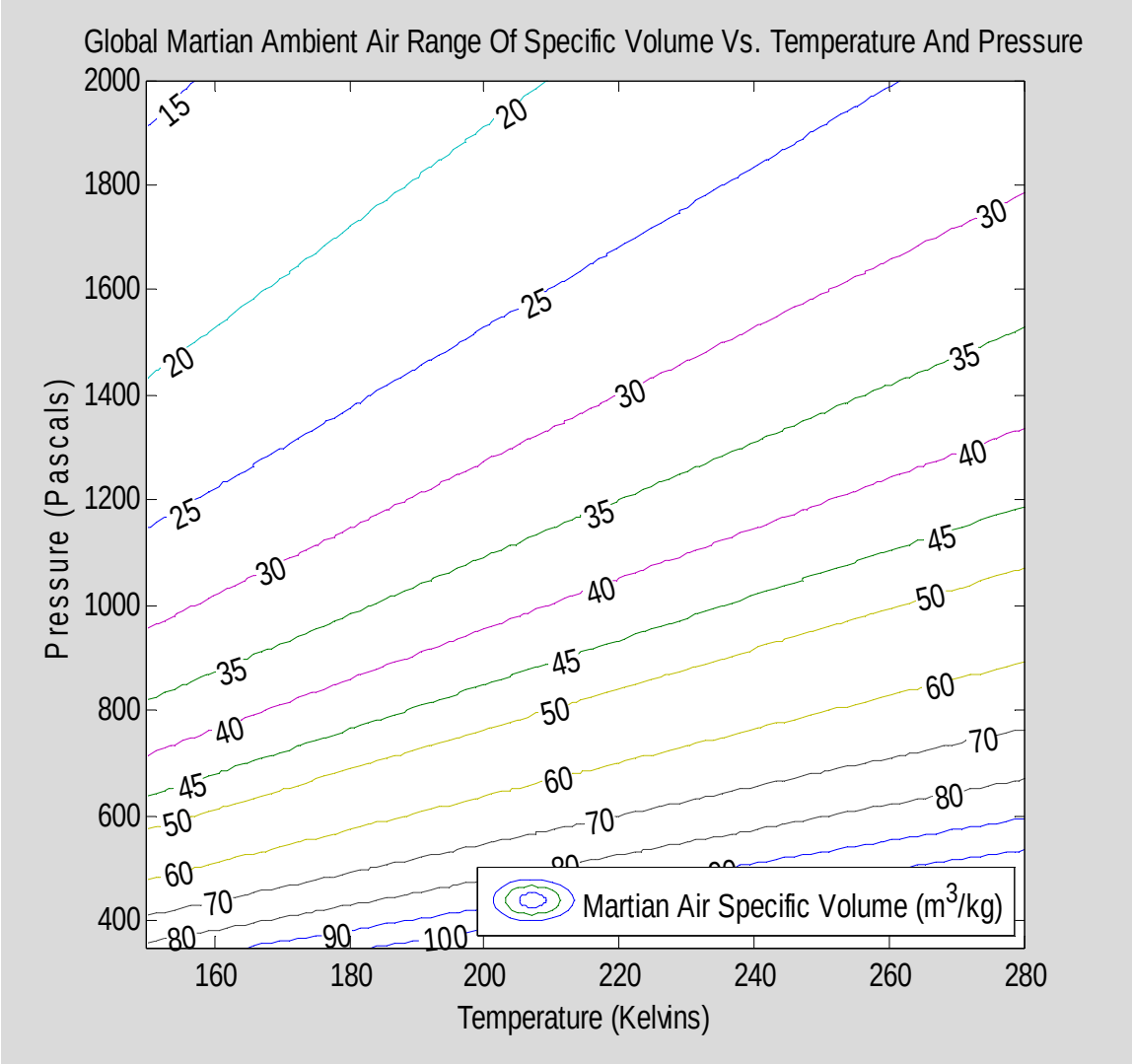


Figure 13-2: Near-Surface Ambient Martian Air Specific Volume, Pressure, and Temperature

independent of gas pressure. They are readily measured thermodynamic property from which other important properties can be determined, and they are also related to each other in a simple way, as follows:

$$c_v(T) = c_p(T) - R \tag{13-2}$$

These two specific heat functions of temperature for MarsMix are plotted in Figure 13-3.

The dimensionless ratio of these specific heats can be important for adiabatic process calculations if it is almost constant. If it is sufficiently close to being a constant over some temperature range, then adiabatic processes in that range can be analyzed by simple algebraic formulae instead of requiring more complicated numerical operations on tabulated functions. Although some authors prefer the symbol, γ, herein the specific heat ratio is denoted as κ(T), i.e.,

$$\kappa(T) = \frac{c_p(T)}{c_v(T)} \tag{13-3}$$

The κ(T) specific heat ratio of MarsMix gas is plotted as Figure 13-4, which also includes for comparison purposes the specific heat ratios of EarthAir gas and of helium gas. The plot reveals a considerable difference among their behaviors. The specific heat ratio for helium is constant while for EarthAir the ratio varies a bit and for MarsMix it varies a lot. Also, the ratio is much larger for helium, intermediate for EarthAir and lowest of all for MarsMix. These differences in specific heat ratio lead to profound differences in the performance of Brayton cycles using the different gases as working fluids.

Internal energy of a perfect gas, u(T), can be determined from its specific heat at constant volume by integration:

$$u_2 - u_1 = \int_{T_1}^{T_2} c_v(T)dT \tag{13-4a}$$

Enthalpy, *h(T)* of a perfect gas can be determined from its specific heat at constant pressure by integration:

$$h_2 - h_1 = \int_{T_1}^{T_2} c_P(T)dT \tag{13-4b}$$

Because of the Eq. 13-2 relation between the specific heats, the enthalpy of a gas differs additively from its internal energy by $RT=pV_{sp}$. In Brayton Cycle analyses, enthalpy is used more often than internal energy. For gases with constant temperature-independent specific heats, the specific heat can be factored out to yield the following:

$$h_2^{(\text{Constant Cp})} - h_1^{(\text{Constant Cp})} = (T_2 - T_1)c_P \tag{13-5}$$

However, for gases with variable specific heats, the best we can do is to numerically tabulate and plot the functions. Enthalpy of MarsMix gas is plotted in Figure 13-5.

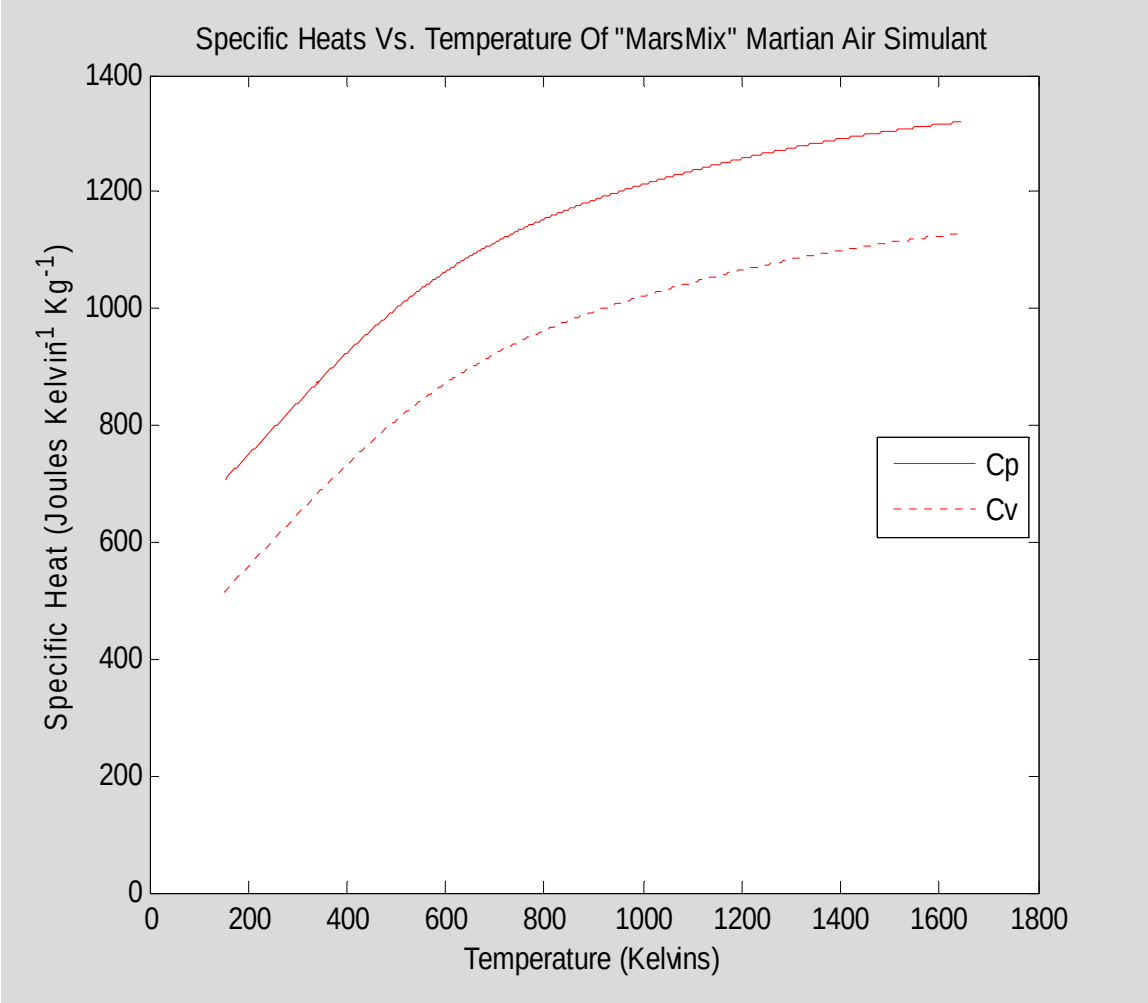


Figure 13-3: Specific Heats of MarsMix Martian Air Simulant

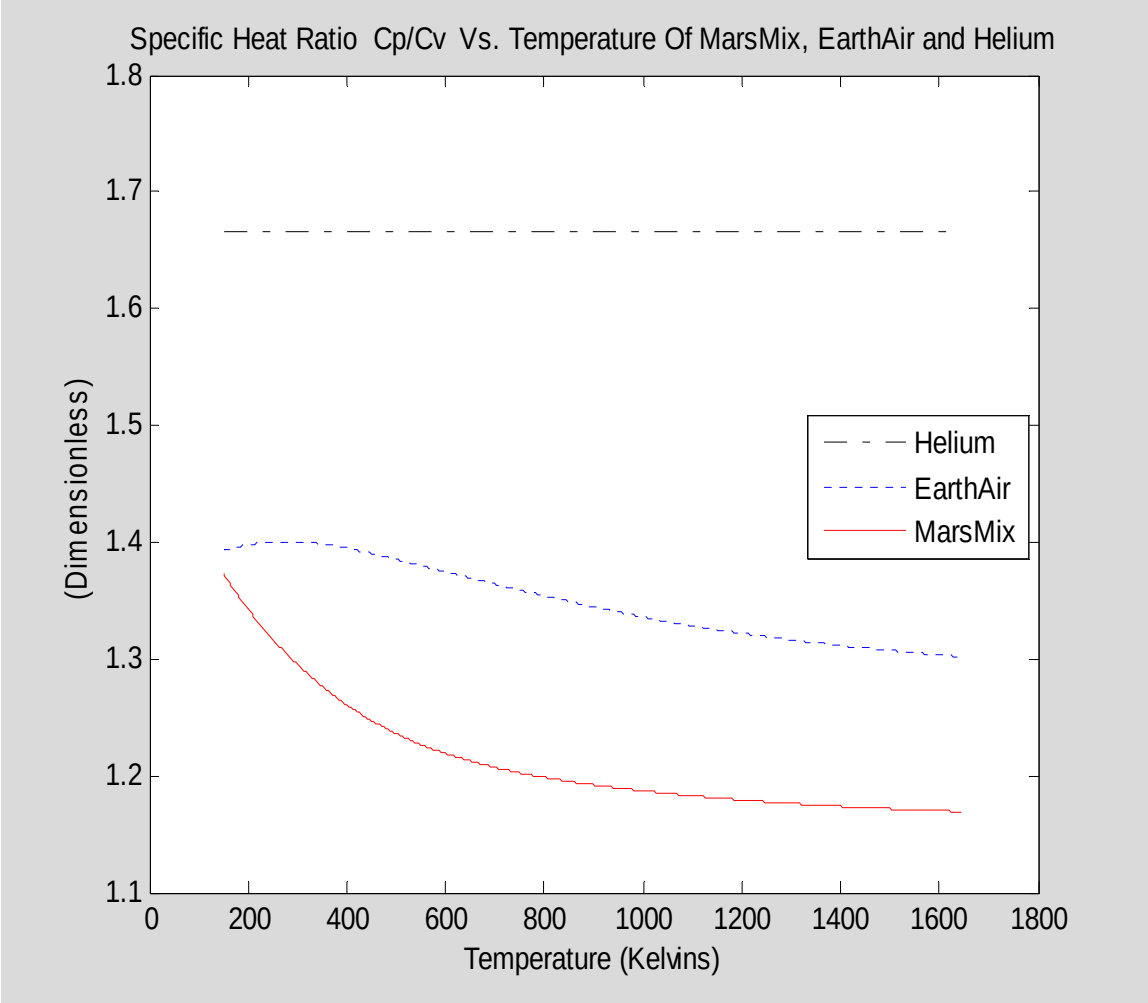


Figure 13-4: Ratio Of Specific Heats of MarsMix, EarthAir, and Helium

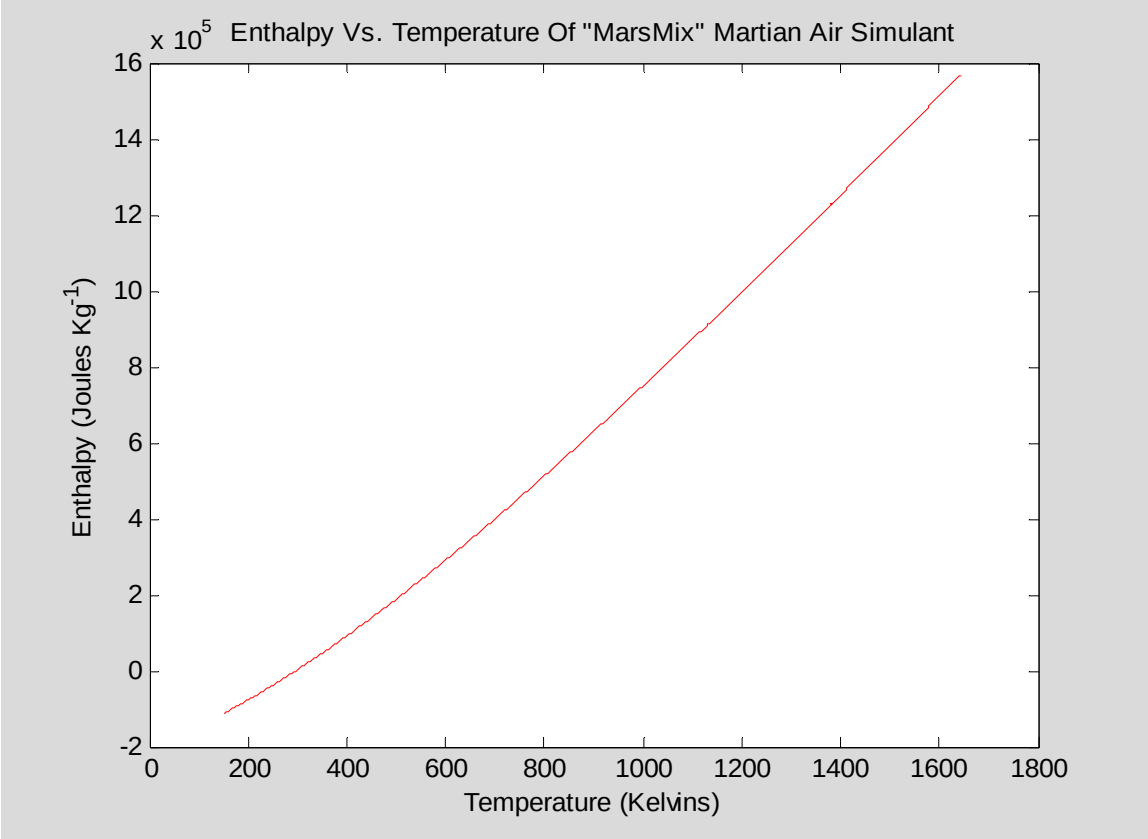


Figure 13-5: Enthalpy of MarsMix Martian Air Simulant

Entropy is a crucially important thermodynamic gas property. Absolute entropy is never needed for Brayton Cycle analyses but entropy differences are used frequently. Entropy differences between states can be calculated using an integral involving specific heat along with a formula involving the gas pressures of the two states.

$$s_2 - s_1 = \int_{T_1}^{T_2} \frac{c_p(T)}{T} dT - R \ln\left(\frac{p_2}{p_1}\right) \tag{13-6}$$

For gases which have a constant temperature-independent value of the specific heat at constant pressure, that constant may be factored out of the integral to yield:

$$s_2^{(\text{Constant Cp})} - s_1^{(\text{Constant Cp})} = c_p \ln\left(\frac{T_2}{T_1}\right) - R \ln\left(\frac{p_2}{p_1}\right) \tag{13-7}$$

However, to simplify numerical application of Eq. 13-6 for gases with variable specific heat, it is conventional to define the following entropy integral function of temperature (which is tabulated in various reference publications for particular gases):

$$S_0[T] \equiv \int_{T_0}^{T} \frac{c_P(T)}{T} dT \tag{13-8}$$

To avoid confusion, in Eq. 13-8 I have invented a notation for non-algebraic tabulated numerical functions in which the function's name appears first, followed by ***square brackets*** containing the function's argument. This notation is used freely in the following analyses. Square brackets will also be used in referring to the *inverse* of a tabulated function, denoted by the function's name followed by a minus one superscript to denote inverse, then followed by square brackets enclosing the argument of the inverse function.

In this notation, Eq. 13-6 can be rewritten using the entropy integral function as follows:

$$s_2 - s_1 = S_0[T_2] - S_0[T_1] - R \ln\left(\frac{p_2}{p_1}\right) \tag{13-9}$$

This Eq. 13-8 entropy integral function for MarsMix gas is plotted as a function of temperature in Figure 13-6.

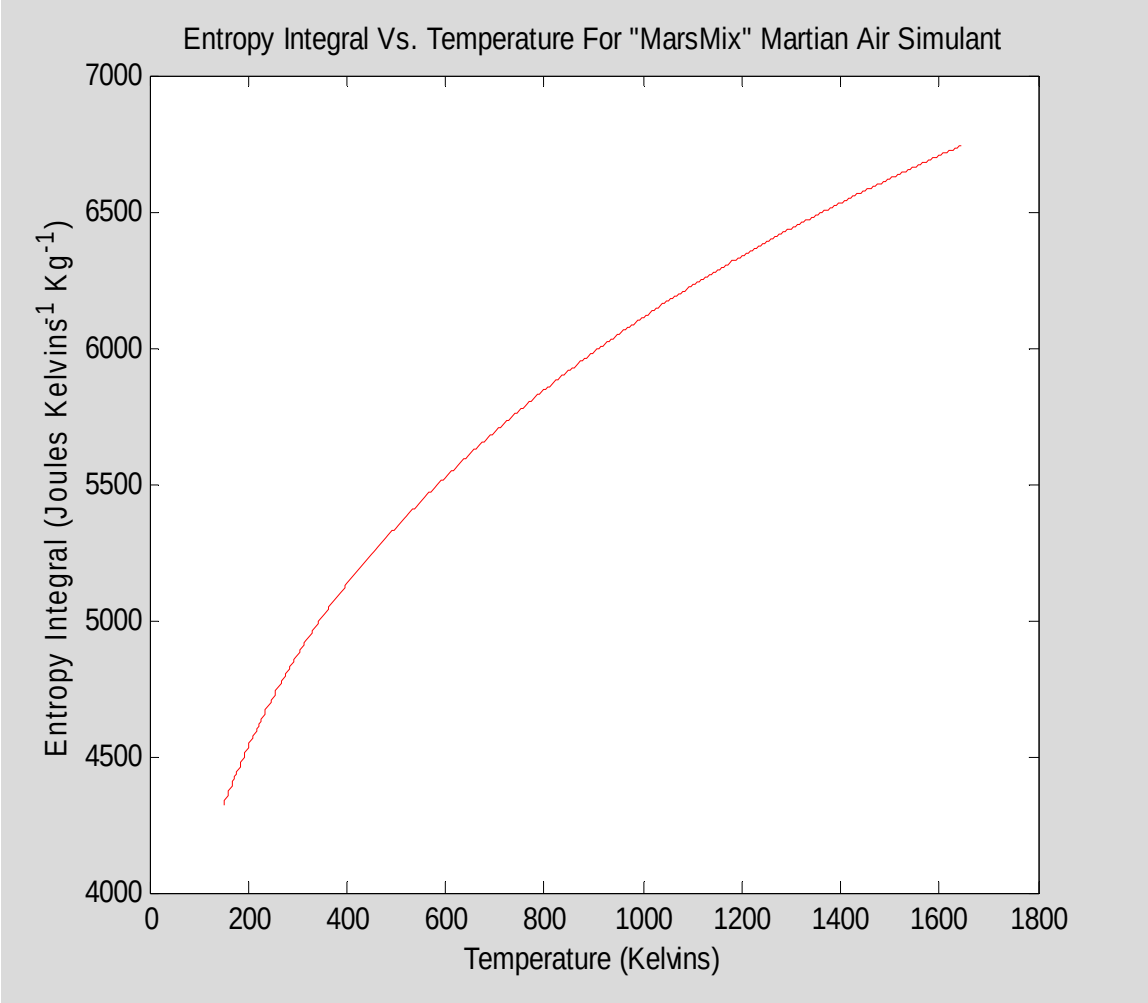


Figure 13-6: Entropy Integral Function of MarsMix Martian Air Simulant

13.3 Ideal Adiabatic Compression Or Expansion Of Martian Air

Equation 13-9 can be solved for the pressure ratio between two states in terms of their entropies and the tabulated entropy integral function, as follows:

$$\frac{p_2}{p_1} = \exp\left(\frac{S_0[T_2] - S_0[T_1] - (s_2 - s_1)}{R} \right) \tag{13-10}$$

If an adiabatic compression process without heat addition takes the gas between those two states then their entropy difference is zero. Thus we can write the pressure ratio for adiabatic compression as follows:

$$\left(\frac{p_2}{p_1} \right)_{\text{adiabatic}} = \exp\left(\frac{S_0[T_2] - S_0[T_1]}{R} \right) \tag{13-11}$$

For gases in which the specific heat ratio is constant over the temperature range, a simpler closed-form algebraic expression results:

$$\left(\frac{p_2}{p_1} \right)_{\text{adiabatic isentropic}}^{\text{constant } c_p} = \left(\frac{T_2}{T_1} \right)^{\left(\frac{\kappa}{\kappa - 1} \right)} \tag{13-12}$$

However, MarsMix specific heats are far from being constant, so Eq. 13-11 is used for adiabatic compressions and expansions in the present design instead of Eq. 13-12.

Gas temperature naturally increases as it is being compressed and, as the adiabatic compression pressure ratio is increased, the associated temperature rise caused by the compression also increases. Figure 13-7 plots vs the compression pressure ratio the calculated temperature of MarsMix gas resulting from adiabatic compression without any heat addition, assuming the gas initially starts at the 215 Kelvins global average temperature of ambient martian air. The maximum plotted compression ratio, 76,173.1, raises the MarsMix gas temperature from 215 Kelvins to 1643 Kelvins. For comparison purposes, Figure 14-6 also plots similar curves for helium and EarthAir, assuming they all start at the same initial temperature.

High compression ratios are desired because they allow high energy conversion efficiencies. But because compression ratios are interrelated with gas temperatures, Brayton cycle compression ratios are fundamentally temperature-limited, either by high temperature constraints imposed by materials limits or by the temperature of the heat source. In this regard, alternative gases are not created equal as thermal conversion working fluids. As shown in Figure 13-7, for any high temperature limit chosen, helium

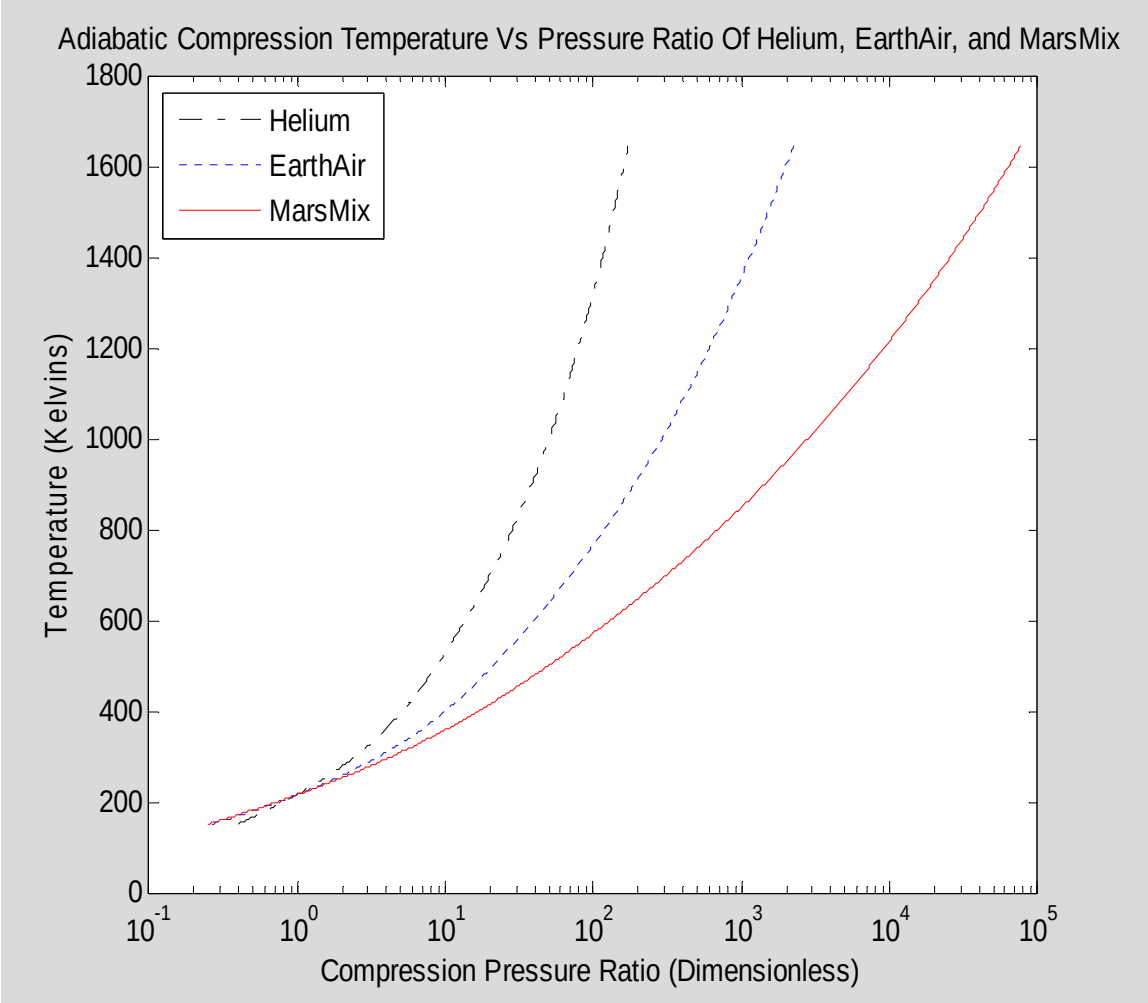


Figure 13-7: Temperature Vs Compression Pressure Ratio Of Helium, EarthAir, and MarsMix

is the most restrictive in limiting the compression ratio. EarthAir allows a higher compression ratio than helium for that same maximum temperature, but MarsMix allows the highest compression ratio of all three gases. This superior feature of martian air is convenient, since the martian ambient air pressure is low.

For MarsMix gas starting at the average ambient air condition of 215 Kelvins temperature and 850 Pascals pressure, the endpoint conditions reachable by adiabatic compression or expansion are given in tems of the tabulated entropy integral function by the following formula

$$p(T) = 850 \exp\left(\frac{S_0[T] - S_0[215]}{R} \right) \tag{13-13}$$

This pressure-temperature relation is plotted in the Figure 13-8 graph, along with global ambientmartain air intake pressure and temperature conditions for comparison purposes.

Adiabatic compression of 850 Pa MarsMix martian air simulant to reach $1.013*10^5$ Pa, thus matching the sea level pressure of Earth's air, only raises its temperature from 215 Kelvins to 589 Kelvins (i.e., 316 C). Heat added from a nuclear reactor heat source can then heat it to considerably hotter temperatures for which available engine materials can function well within their temperature limits. Thus, there is no obvious impediment preventing an Open Brayton Cycle implementation using martian air as its working fluid.

Thermal conversion requires gas *expansion* in addition to compression, from different thermodynamic starting points at higher temperatures. Therefore, Figure 13-9 graphs a family of adiabat pressure vs temperature curves. The curves deviate from straight lines on this type of loglog graph because the specific heats of MarsMix are not constant.

Two adiabat curves are traversed in a simple ideal Open Brayton Cycle (OBC) implementation, one for the compression and the other for the expansion.

13.4 The Ideal Open Brayton Cycle (OBC) Pattern With Martian Air

An alternative to the identification of thermodynamic states by their temperatures and pressures, i.e., the (T, p) plane, is to instead identify them by their specific volumes and pressures, i.e., in the (V_{sp}, p) plane. The same adiabats are replotted in the (T,p) plane in Figure 13-10.

Other pairs of independent thermodynamic variables are frequently used in analyses. Perhaps the most frequent are the (s, T) plane and the (s, h) plane, the latter of which is sometimes called a Mollier diagram. In each of those planes, adiabats, having constant entropy by definition, appear as a set of parallel vertical lines. A special utility of the (s, h) plane with linear axes is that vertical distances along adiabat curves are equal to the compression or expansion work done.

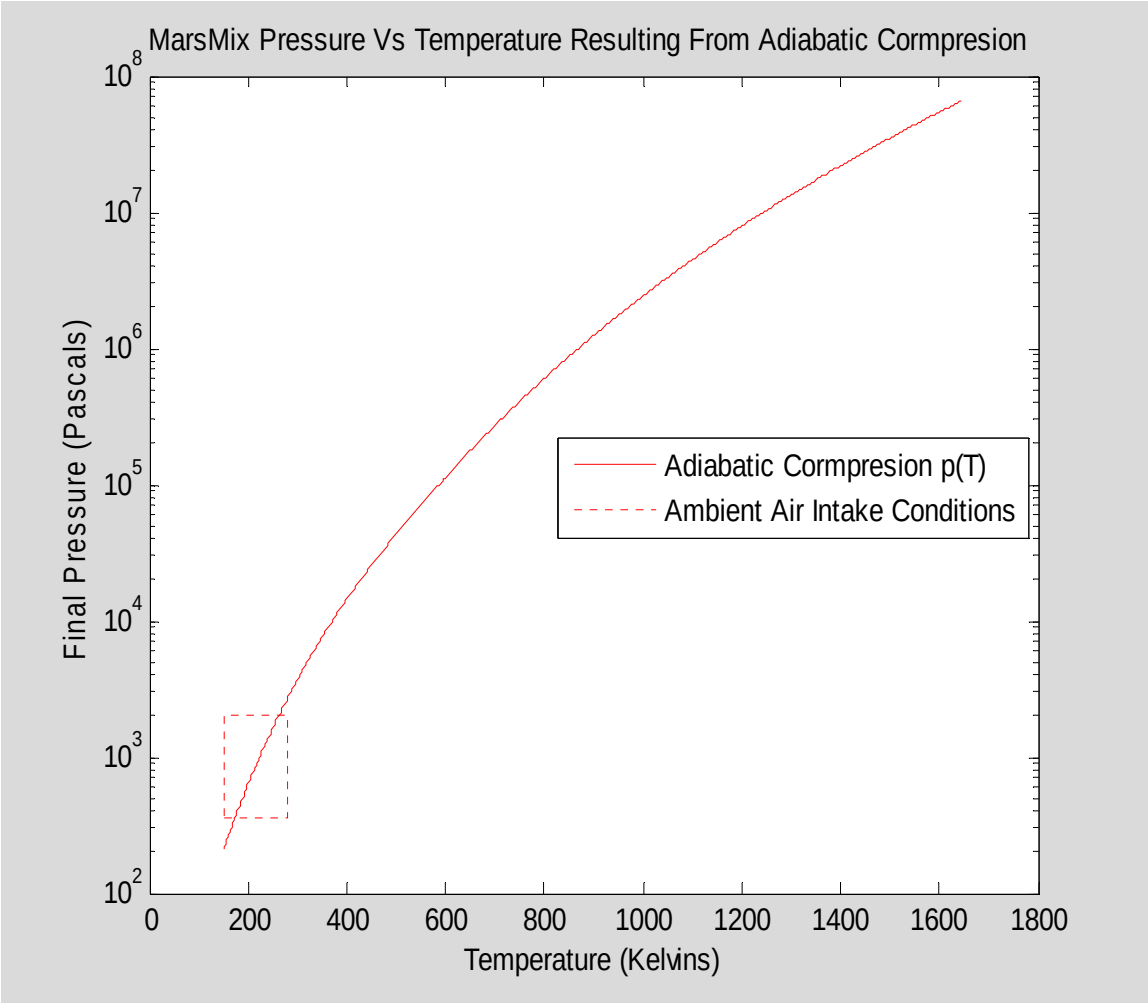


Figure 13-8: Pressure Vs. Temperature Trajectory Of MarsMix In Adiabatic Compression Or Expansion

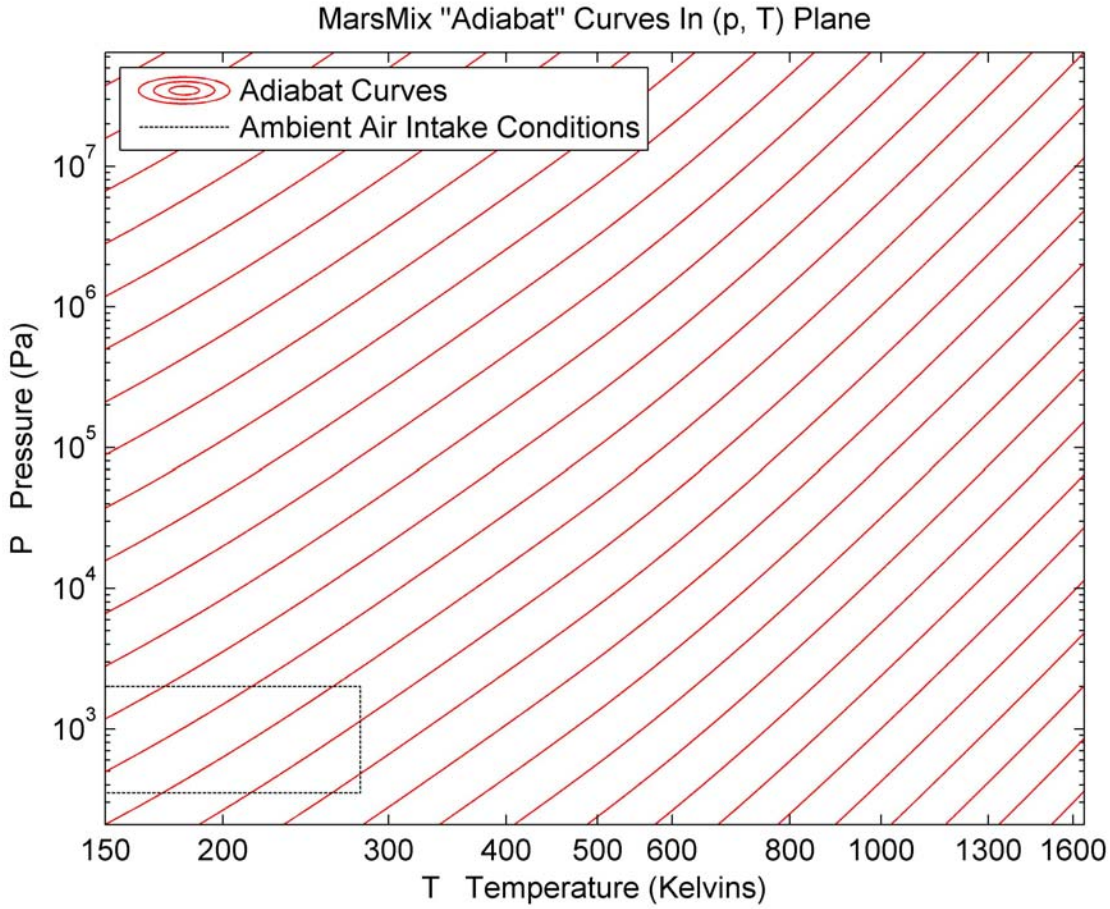


Figure 13-9: Pressure Vs. Temperature Adiabat Trajectory Curves In (T, p) Plane For MarsMix Gas In Adiabatic Compression Or Expansion

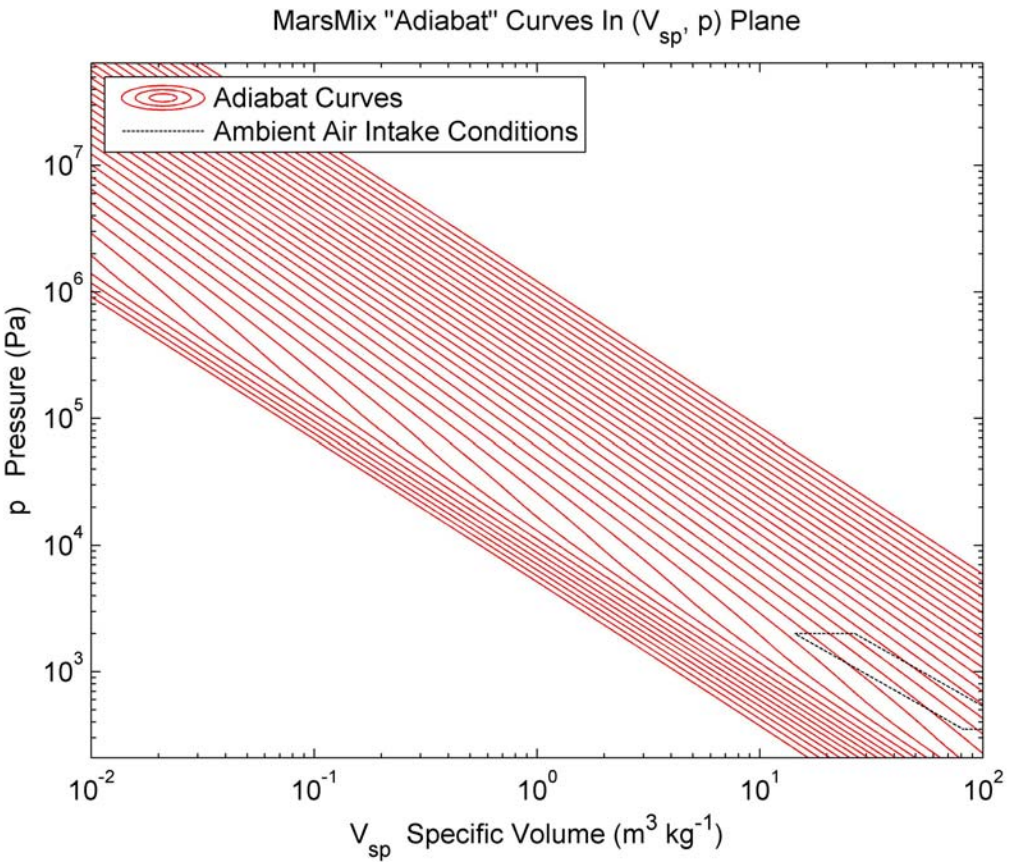


Figure 13-10: Pressure Vs. Specific Volume Adiabat Trajectory Curves Im (T, V_{sp}) Plane For MarsMix Gas In Adiabatic Compression Or Expansion

13.5 Ideal Reversible Constant-Pressure Heat Transfer To Martian Air

The simple nuclear-heated Open Brayton Cycle of Figure 4-1 requires that heat from the reactor be added via a heat exchanger to the compressed air. Constant-pressure (i.e., isobaric) reversible heat addition to the MarsMix gas is modeled in the present section. Within a real heat exchanger there are two departures from ideal reversibility. The first is that a local temperature drop exists between the reactor coolant and the compressed air, needed because heat only flows from hotter to colder masses. Even if a counter-flow heat exchanger is used in order to reduce the average temperature drop in a heat exchanger, it still can be significant. The second departure is that the compressed air flowing through the heat exchanger also suffers a *pressure* drop due to flow friction losses. Each of these two real effects reduces efficiency when the heat exchanger is part of a thermal energy conversion system. Since these effects can in principle be made arbitrarily small by simply increasing without limit the heat exchanger's size and mass, both are ignored in this section.

In the idealized constant-pressure model, the heat added does not match the change in gas internal energy. The ideal gas law, Eq. 13-1, requires that gas density decrease as its temperature increases due to isobaric heat addition. Equivalently, this means increasing specific volume. Even though there is no obvious piston or turbine involved, this implies work energy is being mechanically extracted from the gas. If the specific volume of the gas changes from V_1 to V_2, the work energy removed from the gas is $p(V_2-V_1)$.

However, if the pV_{sp} product of pressure and specific volume is added to the internal energy, u, then the enthalpy, h, is obtained as their sum. It is easy to show that for constant-pressure processes, the heat added must match the change in gas enthalpy.

In either the (T, p) plane or the (V_{sp}, p) plane a constant pressure heat addition process is plotted simply as a horizontal line at the prevailing pressure. The constant-pressure heat addition trajectory curve is more complicated in the (s, T) plane or the (s, h) plane since entropy of the MarsMix gas increases nonlinearly along with its temperature T and enthalpy, h, throughout the heat addition. For gases with constant specific heats, an algebraic formula exists for these curves, in which the temperature ratio due to heat addition at constant pressure is the exponential function of the ratio of the change in entropy to the specific heat at constant pressure. For isobaric heat addition of q heat energy per unit mass to a gas with constant specific heats, the following equations apply:

$$h^{(\text{constant } c_p, \text{ constant p})} = h_0 + q$$

$$T^{(\text{constant } c_p, \text{ constant p})} = T_0 + \frac{q}{c_p}$$

$$s^{(\text{constant } c_p, \text{ constant p})} = s_0 + c_p \ln\left(1 + \frac{q}{c_p T_0}\right) \qquad \text{(13-14(a,b,c))}$$

However, for gases with non-constant specific heats, the (s, T) plane or (s, h) plane curve calculation procedure involves algebraic use of inverse functions of the tabulated thermodynamic properties. For the (s, T) plane, points along a constant-pressure heat addition curve can be calculated iteratively by first choosing a value for q, the heat added per unit mass, then evaluating the following formulae:

$$\begin{aligned} h_{initial} &= h\left[T_{initial}\right] \\ h_{final} &= q + h_{initial} \\ T_{final} &= h^{-1}\left[h_{final}\right] \\ s_{final} &= s_{initial} + S_0\left[T_{final}\right] - S_0\left[T_{initial}\right] \end{aligned} \tag{13-15}$$

For the (s, h) plane, points along a constant-pressure heat addition curve can similarly be found iteratively by first choosing q but then evaluating the following slightly different formulae.

$$\begin{aligned} h_{final} &= q + h_{initial} \\ T_{initial} &= h^{-1}\left[h_{initial}\right] \\ T_{final} &= h^{-1}\left[h_{final}\right] \\ s_{final} &= s_{initial} + S_0\left[T_{final}\right] - S_0\left[T_{initial}\right] \end{aligned} \tag{13-16}$$

Alternatively, the constant-pressure heat addition curves can be found numerically as the level curves of a single function. For the (s, T) plane, they are the level set contour curves of the following function:

$$F_{sT}(s,T) \equiv \exp\left(\frac{S_0\left[T\right] - S_0\left[T_{\text{Reference}}\right] - s}{R}\right) \tag{13-17}$$

while for the (s, h) plane they are the level set contour curves of:

$$F_{sh}(s,h) \equiv \exp\left(\frac{S_0\left[h^{-1}\left[h\right]\right] - S_0\left[T_{\text{Reference}}\right] - s}{R}\right) \tag{13-18}$$

Although these two functions are mathematically different functional mappings, they give numerically identical values. Their level sets are constant pressure curves, i.e. where pressure is given by the following:

$$p = p_{\text{Reference}} F_{sT}(s,T) = p_{\text{Reference}} F_{sh}(s,T) \tag{13-19}$$

These equations generate plots show families of curves representing isobaric heat addition to MarsMix gas. Figure 13-11 shows the heat addition curves in the (s, T) plane while Figure 13-12 shows them in the (s, h) plane. In these plots, the reference state is taken as 215 Kelvins temperature and 850 Pascals pressure, a typical expected ambient air intake condition.

13.6 Ideal OBC Operation Vs. Compression Ratio & Max Temperature

The simple version of the ideal Open Brayton Cycle compresses atmospheric air in a compressor, then heats it in a heat exchanger, and then expands it back to atmospheric pressure in a turbine. The designer cannot choose the gas conditions for the incoming air, but instead must accept and work with the air temperature and pressure conditions prevailing in the immediate vicinity of the engine's air intake, whatever they may be. The designer can choose one extensive parameter setting the engine's scale, either the air's mass flow rate or the heat source power or the output shaft power, all of which are directly related to each other. However, the designer can only choose two fundamental intensive parameters, i.e.,

(1) the compression (pressure) ratio, and
(2) the maximum temperature reached by the compressed air after heat addition.

Performance is always better with larger temperature differences, so the maximum temperature (i.e., the second parameter) is invariably chosen to be as high as is feasible consistent with temperature constraints imposed by the limited temperature of the ultimate heat addition heat source or by the temperature-dependent strength limitations of engine materials. Thus, the second parameter is not a fully free design parameter to be chosen by the designer. This leaves only the first parameter, the compression ratio, as the fundamental free parameter to be chosen by the designer.

Adopting the nomenclature of Figure 4-1, denote the engine air intake conditions (temperature, pressure, etc.) by using the subscript, 0. The compressor outlet conditions are denoted by the subscript, 2, the heat exchanger outlet by subscript, 4, and the exhaust discharge by the subscript, 5. Denote the air mass flow rate, the heat addition per unit time, and the shaft output power (i.e., work per unit time) by, respectively

$$\dot{m},\ \dot{q},\ P_{\text{out}} \tag{13-20}$$

Then, for an Open Brayton Cycle with perfectly efficient, idealized components, the heat transfers in the heat exchanger and the adiabatic work of the compressor and turbine are each balanced by matching enthalpy changes:

$$\begin{aligned} \left(h[T_4] - h[T_2]\right)\dot{m} &= \dot{q} \\ \left(h[T_0] - h[T_2] + h[T_4] - h[T_5]\right)\dot{m} &= P_{\text{out}} \end{aligned} \tag{13-21(a, b)}$$

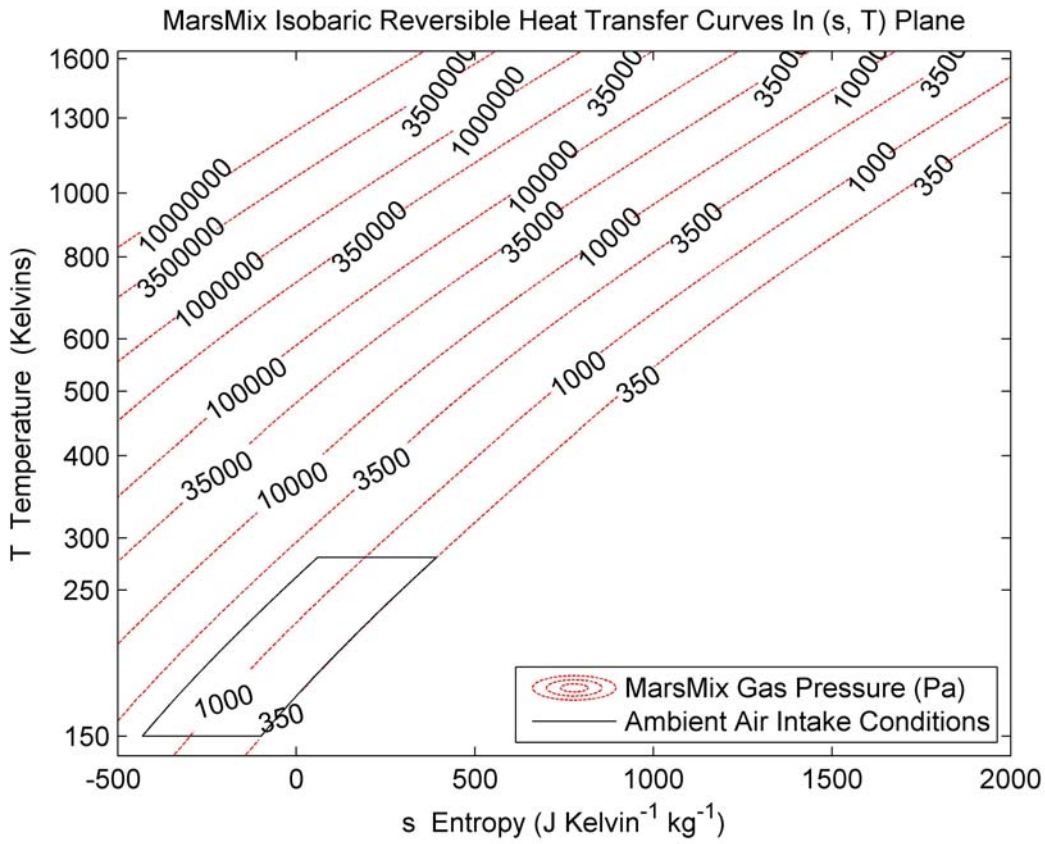


Figure 13-11: Isobaric Reversible Heat Transfer Curves In (s, T) Plane

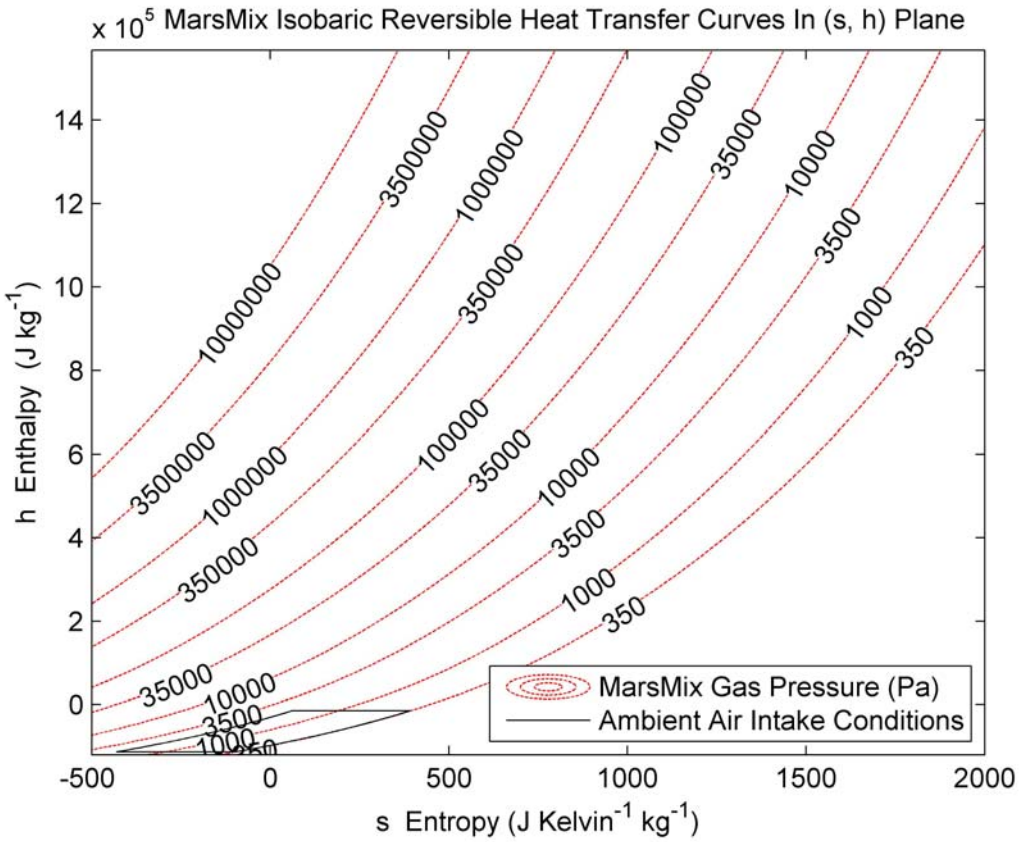


Figure 13-12: Isobaric Reversible Heat Transfer Curves In (s, h) Plane

There are two commonly considered performance measures for an ideal engine, i.e., the energy efficiency and the specific power. The energy efficiency, η, is a dimensionless number, the fraction of the heat energy which is converted to output shaft power, i.e.,

$$\eta \equiv \frac{P_{out}}{\dot{q}} = \frac{h[T_0] - h[T_2] + h[T_4] - h[T_5]}{h[T_4] - h[T_2]} \tag{13-22}$$

The specific power measure of engine performance is intended to quantify the output shaft power per unit of gas flow through the engine. It is a fundamentally *dimensional* quantity since shaft power and gas flow are measured in different units. For instance, if we measure shaft power in watts and gas flow in kilograms per second then the specific power calculated directly as their ratio carries the units, joules per kilogram.

It is conventional for some authors to make the specific work appear dimensionless by further dividing the ratio of output power to mass flow by some fixed number of joules per kilogram, but that fixed number is always arbitrary and bears little or no relation to the engine being analyzed. For example, it is common to divide by the c_pT_0 product where T_0 is the absolute temperature of the intake air. This simplifies formulas, particularly if the working gas has constant specific heats, but other normalization divisors are also used. Since this practice seems unrelated to the actual analysis of engine performance, it is avoided here except where explicitly stated. Specific work is herein referred to in joules per kilogram, and is defined as follows:

$$w_{sp} \equiv \frac{P_{\text{out}}}{\dot{m}} = h[T_0] - h[T_2] + h[T_4] - h[T_5] \tag{13-23}$$

13.6.1 Performance Analysis Assuming Constant Specific Heats & Ideal Components

For a working gas with constant specific heats, analysis of the ideal Open Brayton Cycle assuming no losses in compressor, heat exchanger or turbine is particularly simple and results in closed-form algebraic formulae. Derivations of these formulae are presented in texts on turbomachines. Since the design must accept air temperature at its prevailing temperature, T_0, and since the high temperature, T_4, is usually fixed by heat source or materials issues identified in advance, it is conventional [Decher 1994 , 243-301] to define their temperature ratio as a fixed parameter characterizing the specified design problem:

$$\tau \equiv \frac{T_4}{T_0} \tag{13-24}$$

For example, if a gas at T_0=215 Kelvins were to be compressed and then heated to a maximum temperature of T_4=1290 Kelvins, then τ=1290/215=6. (Note for this example that 215 Kelvins was chosen as the average temperature of martian air. The 1290 Kelvins value was chosen arbitrarily for no particular reason, but seems low enough to permit various practical heat transfer temperature drops without exceeding the 1643 Kelvins materials limit discussed in Chapter 4 for molten lithium in long term sustained contact with refractory metals such as tungsten. Those temperature drops can be made arbitrarily small only by increasing without limit the mass of heat exchange equipment.)

The compression pressure ratio is the parameter chosen by the designer. It is conventional to also calculate the ratio of gas temperatures before and after adiabatic compression or expansion. In this constant specific heats case, this compression temperature ratio is related to the compression pressure ratio through a simple formula:

$$\tau_c \equiv \frac{T_2}{T_0} = \left(\frac{p_2}{p_0} \right)^{\left(\frac{\kappa-1}{\kappa} \right)} \tag{13-25}$$

Because of this relation involving the specific heat ratio, κ, the designer may equivalently choose a value for the compression temperature ratio instead of the compression pressure ratio. The τ_c value chosen is limited by the need to not exceed the specified maximum temperature, T_4, therefore:

$$1 < \tau_c < \tau \tag{13-26}$$

After substitutions into Eqs 13-22 and 13-23 followed by some algebraic simplification, the two performance measures are expressed by the following simple formulae:

$$\eta \equiv 1 - \frac{1}{\tau_c} \tag{13-27}$$

$$w_{sp} = c_p T_0 \left(\frac{\tau}{\tau_c} \right) (\tau_c - 1) \tag{13-28}$$

Using the τ=6 case as an illustrative example, Figure 13-13 plots these two performance measures versus the compression temperature ratio τ_c to be chosen by the designer. The plotted specific work function is normalized to the C_pT_0 product.

It can be helpful to view a cross-plot of these same two performance measures, with τ_c suppressed. This is provided in Figure 13-14. As shown in Figures 13-13 and 13-14, the highest energy conversion efficiency is obtained by setting the compression temperature ratio to almost

$$\tau_c = \tau \tag{13-29}$$

which in this example is 6. To clarify what this means, it is illustrated here with actual

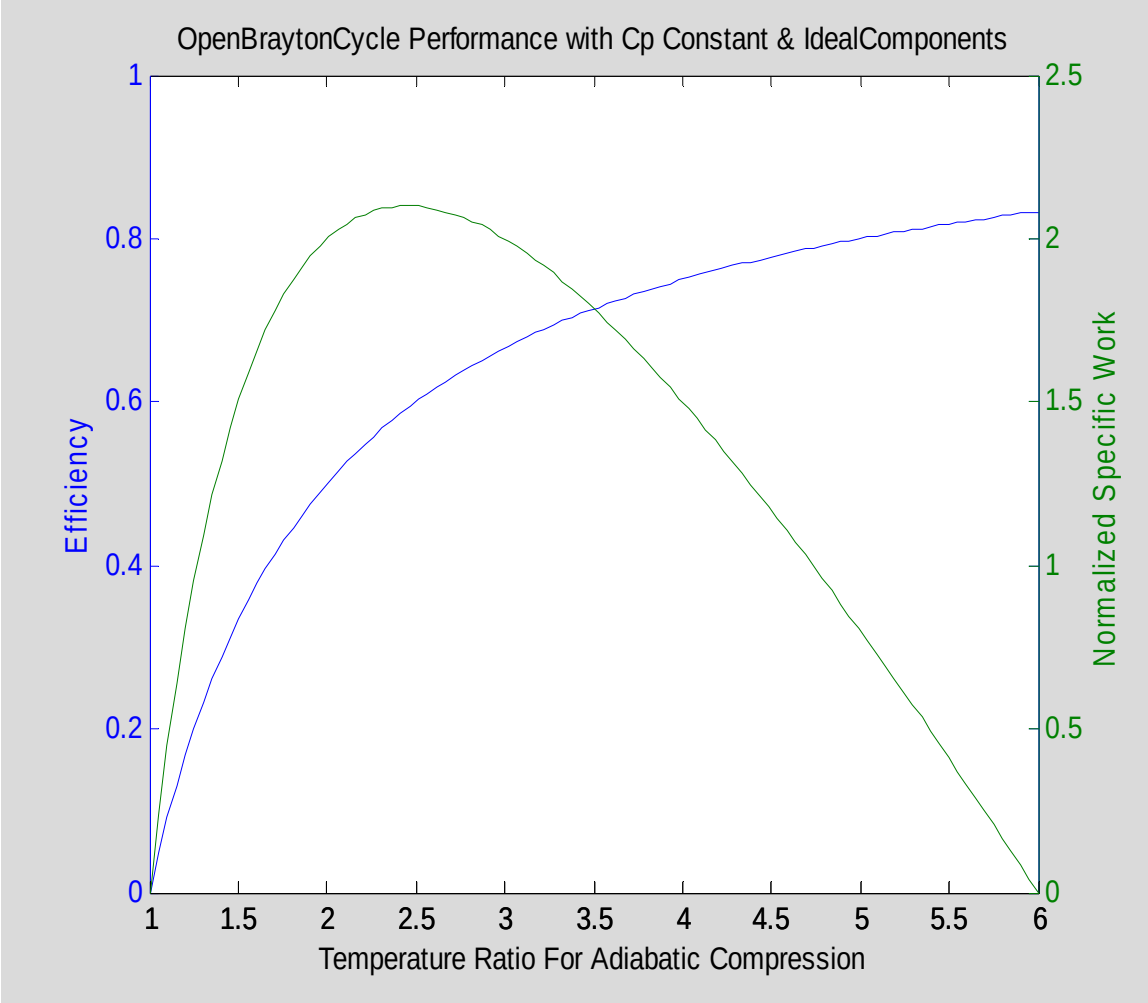


Figure 13-13: Efficiency and Normalized Specific Work Performance Measures For Open Brayton Cycle Using Constant Specific Heats Gas And Ideally Perfect Components

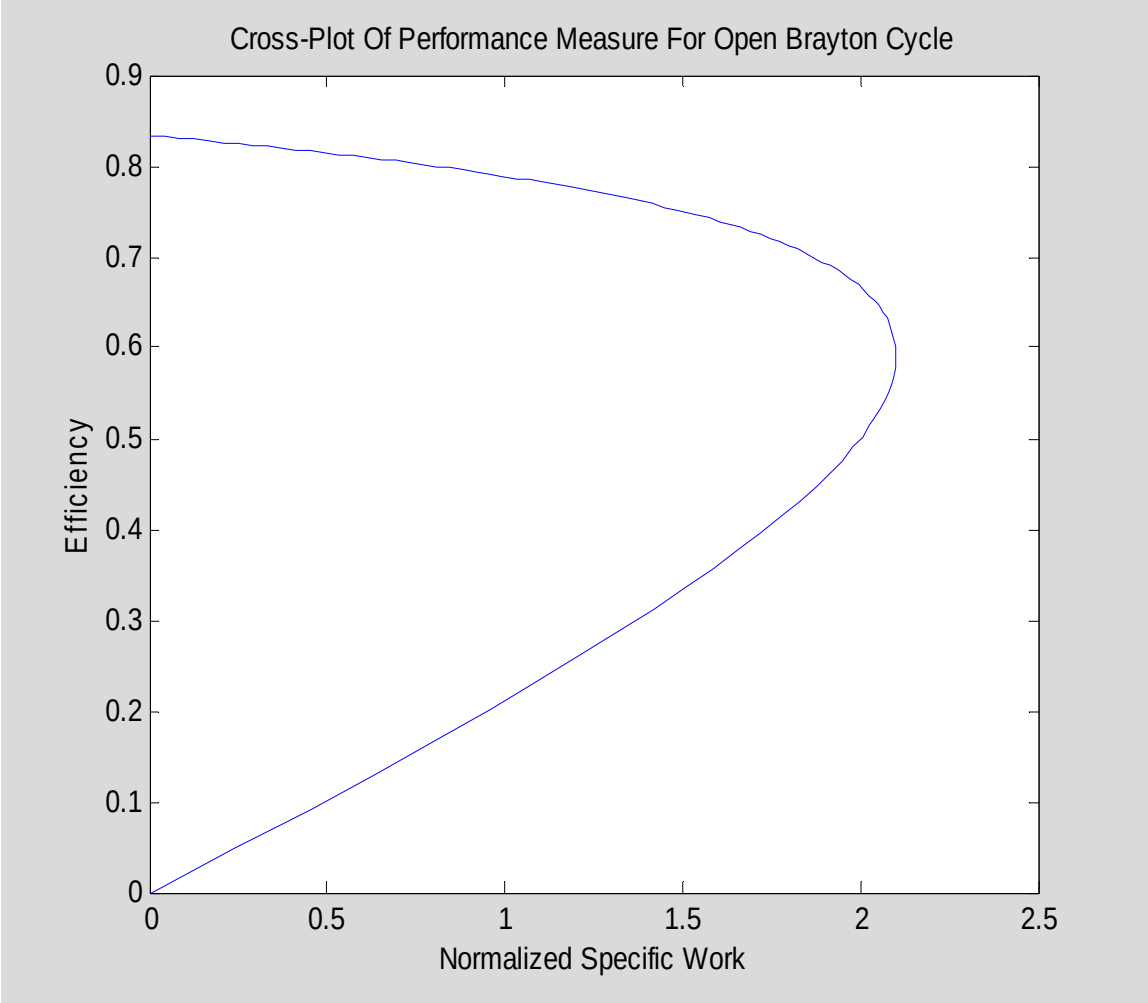


Figure 13-14: Cross-Plot Of Normalized Specific Work Vs Efficiency For Open Brayton Cycle Using Constant Specific Heats Gas And Ideally Perfect Components

temperatures in an example. If T_0 were 215 Kelvins and T_4 were 1290 Kelvins, this corresponds to compressing the gas with such a large compression ratio that it emerges from the compressor at almost 1290 Kelvins, say at 1289 Kelvins. The subsequent heat addition in a heat exchanger would then raise its temperature the rest of the way to 1290 Kelvins, after which a turbine expands it back to its original pressure. The resulting energy conversion efficiency, which it would yield if compressor, heat exchanger, and turbine components were ideally efficient, is about 83%. However, very little heat would be added to raise gas temperature from 1289 to 1290 Kelvins, so very little output shaft power would result. As shown in Figure 13-12, the specific work for this design choice is essentially zero, so this design seems inappropriate in spite of its high energy conversion efficiency. Very much equipment would produce almost zero output power.

Specific power, on the other hand, displays an internal maximum. Elementary calculus applied to Eq. 13-28 shows that the maximum value of the specific work function occurs at $\tau_c=\tau_c^*$ where

$$\tau_c^* = \sqrt{\tau} \tag{13-30}$$

It then follows that the maximum specific work is given by:

$$w_{sp}^* = C_p T_0 \left(\tau - \sqrt{\tau}\right) \tag{13-31}$$

which for the $\tau=6$ case is about $3.55 C_p T_0$. The energy conversion efficiency of the maximum specific work design is

$$\eta^* = 1 - \frac{1}{\tau_c^*} = 1 - \frac{1}{\sqrt{\tau}} \tag{13-32}$$

which for the $\tau=6$ case is about 59%. Thus, if T_0 were 215 Kelvins and T_4 were 1290 Kelvins, then the maximum specific work design would adiabatically increase the gas temperature to T_2=527 Kelvins=254 C by compressing it, after which heat addition would finish heating the gas to 1290 Kelvins before it is expanded in a turbine.

To make this more concrete, suppose the gas were helium, a gas with constant specific heats whose ratio is $\kappa=5/3=1.6667$. We apply Eq. 13-12 and find for the $\tau=6$ case, that the helium compressor pressure ratio for the maximum specific work design is

$$\left(\frac{p_2^*}{p_0}\right)_{(\text{constant } c_p)} = \tau^{\left(\frac{\kappa}{2(\kappa-1)}\right)} = 6^{\left(\frac{5/3}{2(5/3-1)}\right)} = 9.39 \tag{13-33}$$

These formulae do not apply to the martian air because its specific heats vary with temperature. As shown in Figure 13-4, the calculated ratio of specific heats for MarsMix varies throughout the range of expected internal engine temperatures from about 1.38 at low temperatures to about 1.18 at high temperatures.

On the other hand, it is interesting to notice that the optimum compressor temperature ratio for a design maximizing specific work does not depend on the specific heats in any way. The Eq. 13-30 formula was derived under the assumption that the specific heats do not vary with temperature, but it gives the same optimized design value for τ_c regardless of whether the specific heat ratio is large or small.

If the MarsMix gas were further approximated as having constant specific heats with an intermediate ratio of, say, κ=1.25, then these formulas could be applied. Assuming intake air at T_0=215 Kelvins and maximum internal gas temperature of T_4=1290 Kelvins, then to maximize specific work the optimum temperature of gas emerging from the compressor would again be T_2=527 Kelvins. However, the MarsMix compressor pressure ratio for the maximum specific work design as determined from Eq. 13-12 would then be about 88.2 for the same compressor temperature ratio. This result gives an order of magnitude estimate, but should not be considered accurate since the specific heats are actually variable. An analysis of the variable specific heats case valid for MarsMix gas developed as part of the present thesis work appears in the next subsection.

In the case of the present nuclear engine design for Mars surface vehicles, neither the efficiency nor the specific work are appropriate performance measures. Instead, the important engineering goal should be to minimize the total mass of the entire engine given a fixed value of its output shaft power. That total mass includes the nuclear reactor, the radiation shielding, the energy conversion system, and also ancillary structures needed for mobility (e.g., wheels). It is obviously not possible to optimize total mass without considering details of component designs. The usefulness of the efficiency and the specific work as performance measures is precisely that they can be determined without considering any real component design details. Efficiency should be made high since at constant specified output shaft power a higher efficiency implies less massive radiation shielding. Specific work should be made high since at constant specified output shaft power a higher specific work implies less airflow and its associated massive air handling equipment. It is clear that both efficiency and specific work are important but their relative importances cannot be determined without a further tradeoff study based on component design details. However, even without that study it is clear that the optimum design must lie somewhere between the maximum efficiency point and the maximum specific work point, i.e., in the upper portion of the cross-plotted curve in Figure 13-14.

13.6.2 MarsMix Performance Analysis: Variable Specific Heats & Ideal Components

To analyze ideal Brayton cycle performance with a gas whose variable specific heats are tabulated functions of temperature, the methods developed herein use the function and inverse function notation advanced earlier. With variable specific heats, the compression temperature ratio in the idealized compressor is not so succinctly expressed algebraically in terms of the compression pressure ratio and in general does not match the expansion temperature ratio developed in the idealized turbine even though its pressure ratio does match. Denoting the cycle's designed compression pressure ratio as:

$$\left(\frac{p_2}{p_0}\right)\equiv\pi_c \tag{13-34}$$

then Eq 13-11 can be solved for unspecified cycle temperatures in terms of specified intake temperature, maximum gas temperature, and the cycle's pressure ratio, by using the tabulated entropy integral function, $S_0[\bullet]$ and its inverse function, $S_0^{-1}[\bullet]$:

$$T_2 = S_0^{-1}\left[S_0[T_0]+R\ln\pi_c\right] \tag{13-35}$$

The turbine's expansion pressure ratio matches the compressor's, so:

$$T_5 = S_0^{-1}\left[S_0[T_4]-R\ln\pi_c\right] \tag{13-36}$$

Substituting these into Equations 13-22 results in the following:

$$h[T_4]-h\left[S_0^{-1}\left[S_0[T_0]+R\ln\pi_c\right]\right]=\frac{\dot{q}}{\dot{m}} \tag{13-37}$$

$$h[T_0]-h\left[S_0^{-1}\left[S_0[T_0]+R\ln\pi_c\right]\right] + h[T_4]-h\left[S_0^{-1}\left[S_0[T_4]-R\ln\pi_c\right]\right]=\frac{P_{\text{out}}}{\dot{m}}\equiv w_{\text{sp}} \tag{13-38}$$

Given specified values of the endpoint temperatures, T0 and T4, and given any specified value of an Open Brayton Cycle design's compression ration, π_c, Eqs. 13-37 and 13-38 use the lookup table function notation to specify how to calculate the rates of heat input per unit mass flow and the output shaft power per unit mass flow (i.e., the specific work). These formulas can be combined into Eq 13-22 to obtain the efficiency, as follows:

$$\eta = \frac{h[T_0]-h\left[S_0^{-1}\left[S_0[T_0]+R\ln\pi_c\right]\right]+h[T_4]-h\left[S_0^{-1}\left[S_0[T_4]-R\ln\pi_c\right]\right]}{h[T_4]-h\left[S_0^{-1}\left[S_0[T_0]+R\ln\pi_c\right]\right]} \tag{13-39}$$

The various lookup table functions and inverse functions of Eqs. 13-38 and 13-39 were implemented through interpolation in MATLAB for MarsMix gas using the properties tables developed in Appendix C. Assuming an intake temperature of 215 Kelvins and a maximum internal gas temperature of 1290 Kelvins, a cross-plot of the resulting calculated values of specific work and efficiency appears in Figure 13-15. This appears qualitatively similar to the constant specific heats case plotted in Figure 13-13, but was generated using the look-up table functions for MarsMix gas. Special points on the curve must be found by direct numerical calculation instead of relying on analytic methods such as differentiation of algebraic forms.

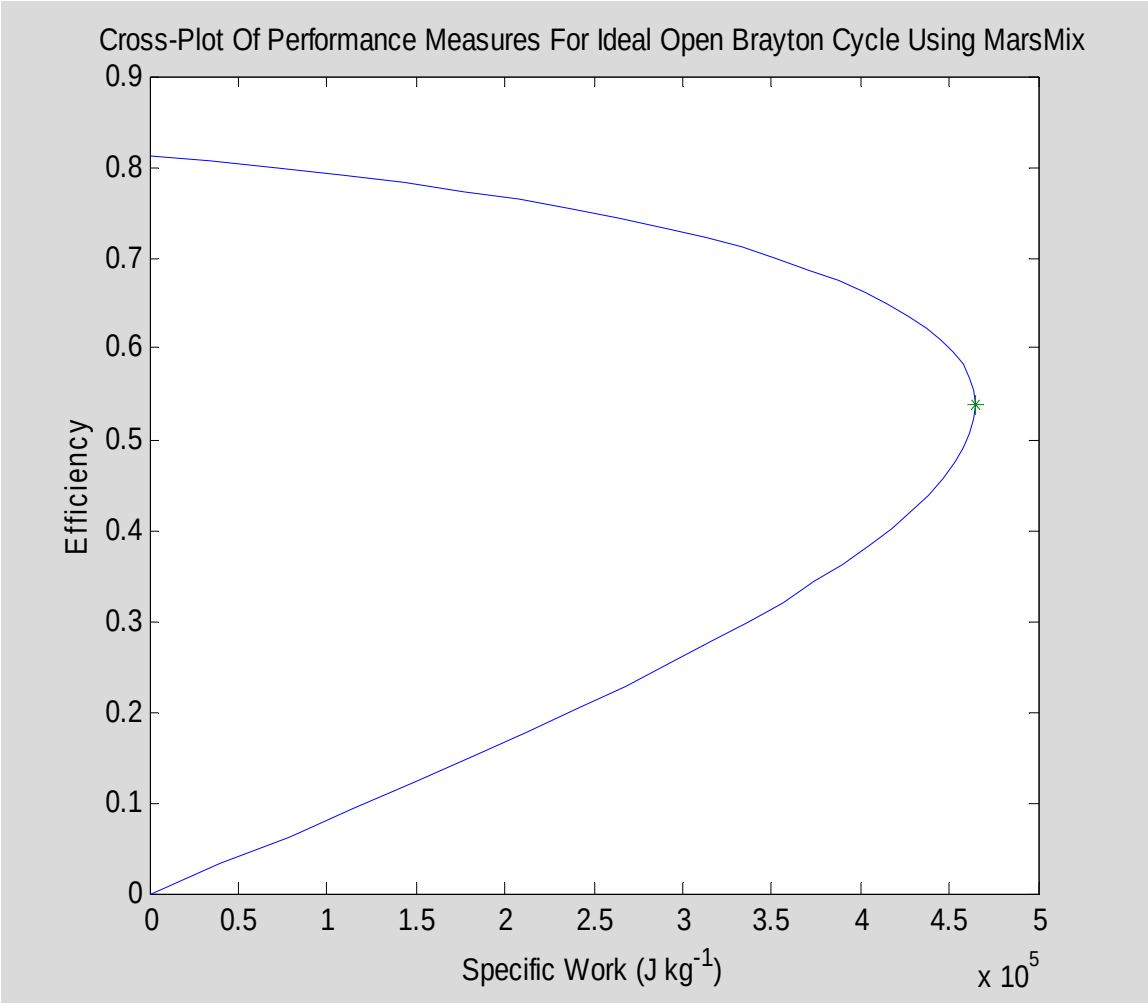


Figure 13-15: Specific Work Vs Efficiency For Ideal Open Brayton Cycle Using Variable Specific Heats MarsMix Gas, Assuming T0=280 Kelvins and T4=1400 Kelvins

The design point for maximum specific work, marked with an * in Figure 13-15, has a specific work value of $4.80*10^5$ joules per kilogram. To be perfectly clear, this implies that an engine with ideal components at this design point which is processing 1 kilogram per second of the martian air will develop an output shaft power of 480 kilowatts. In conventional US units, 480 kilowatts is 643 horsepower. In order to correspond to Engine Model Number 3 of Table 4-2, this design point has been scaled to deliver 316 horsepower from its output drive shaft in the Table 13-1 list of design parameters for this optimized maximum specific work case. Figures 13-16, 13-17, and 13-18 show the cycle trajectory in respectively the (s,T) plane, the (s, h) plane, and the (T,p) plane.

Because the Figure 13-15 cross-plot curve is locally vertical at the maximum specific work design point, some nearby design points have almost as much specific work but significantly higher efficiencies. These nearby design points could be attractive if at constant output shaft power the boost in efficiency reduces the mass of necessary radiation shielding more than it increases the mass of necessary air handling equipment.

13.7 The Open Brayton Cycle With Lossy Components

13.7.1 OverviewDiscussion

The previous sections considered systems with ideal components which operate without component energy losses. Their calculated efficiency values (e.g., up to 83%) were considerably higher than is common in real systems. For *real* Open Brayton Cycle systems of the simple configuration of Figure 4-1, efficiencies are often in the range from 20% to 35%. Thus, the parameters of Table 13-1, while theoretically correct for idealized systems, are quite inaccurate for practical systems.

Some *closed* Brayton cycle systems have reached and even slightly exceeded 50% efficiencies by employing additional heat exchangers to couple some of their exhaust heat back to regeneratively help heat the compressor outlet gas flow. That can help for some design parameter ranges. The higher efficiency performance of such systems is not surprising since their analysis is quite different. However, they are irrelevant here since their special additional heat exchangers are too massive for mobile systems. The large discrepancy between efficiencies of ideal and real OBC systems is due to losses in their components. No real compressor or turbine operates with 100% efficiency, and pressure losses in a heat exchanger due to flow friction effects cannot be eliminated.

13.7.2 Models Of Lossy Turbomachines

The turbomachine community conventionally defines both an adiabatic efficiency and a polytropic efficiency for fluid dynamic machines, and each of these in turn is defined in slightly different ways for compressors and for turbines respectively, so that quoted values never exceed 100%. That tradition results in four slightly different definitions in practice, but all of them have proven useful in the past.

Table 13-1: Parameters Of An Optimized Maximum Specific Work Design Point For The Open Brayton Cycle Using Martian Air, Assuming Ideally Efficient Components

Parameter	Symbol	Value
Intake Air Temperature	T_0	**215 Kelvins**
Intake Air Pressure	p_0	**850 Pa**
Intake Air Mass Flow Rate	**mdot**	**0.491 kg/sec**
Intake Air Volume Flow Rate	**Q**	**23.7 m^3/sec**
Compressor Shaft Power	W_c**dot**	**(175.5 kw)**
Compression Pressure Ratio	π_c	**132.57**
Compressor Outlet Pressure	p_2	**112,685 Pa**
Compressor Outlet Temperature	T_2	**601 Kelvins**
Reactor Thermal Power (rate of heat addition)	**qdot**	**402.3 kW**
Heat Exchanger Outlet Temperature At Turbine Inlet	T_4	**1290 Kelvins**
Heat Exchanger Outlet Pressure At Turbine Inlet	p_4	**112,685 Pa**
Turbine Shaft Power	W_E**dot**	**411.4 kW**
Turbine Outlet Pressure (Exhausted To Atmosphere)	p_5	**850 Pa**
Turbine Outlet Temperature (Exhausted To Atmosphere)	T_5	**583.1 Kelvins**
Net Heat Exhausted In Air Flows		**166.4 kW**
Net Output Shaft Power (SI units)	**P**	**235.9 kW**
Net Output Shaft Power (US customary units)	**P**	**316 horsepower**
Efficiency	η	**58.6%**
Specific Work	**w**	**$4.80*10^5$ J kg^{-1}**

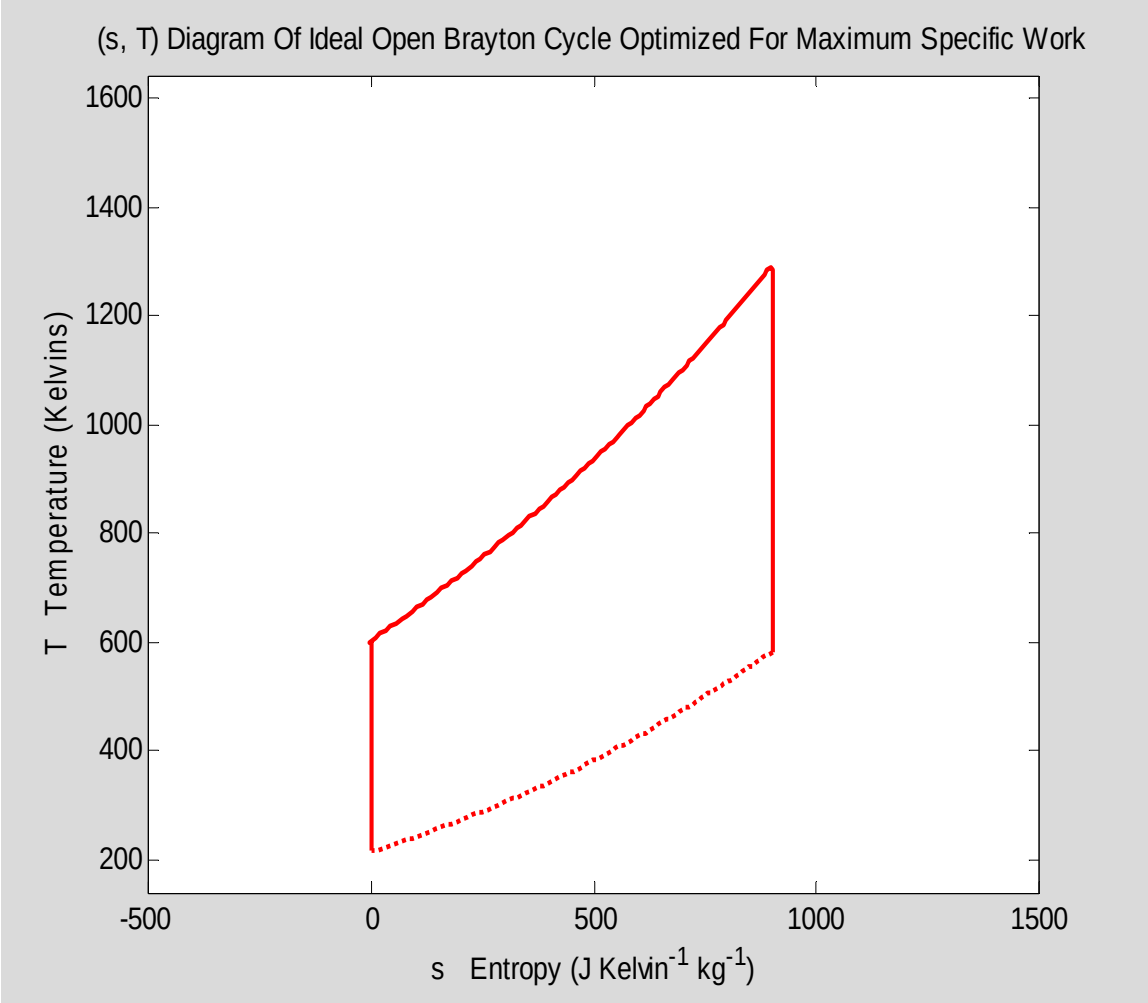


Figure 13-16: (s, T) Plane Ideal MarsMix Open Brayton Cycle Optimized For Maximum Specific Work

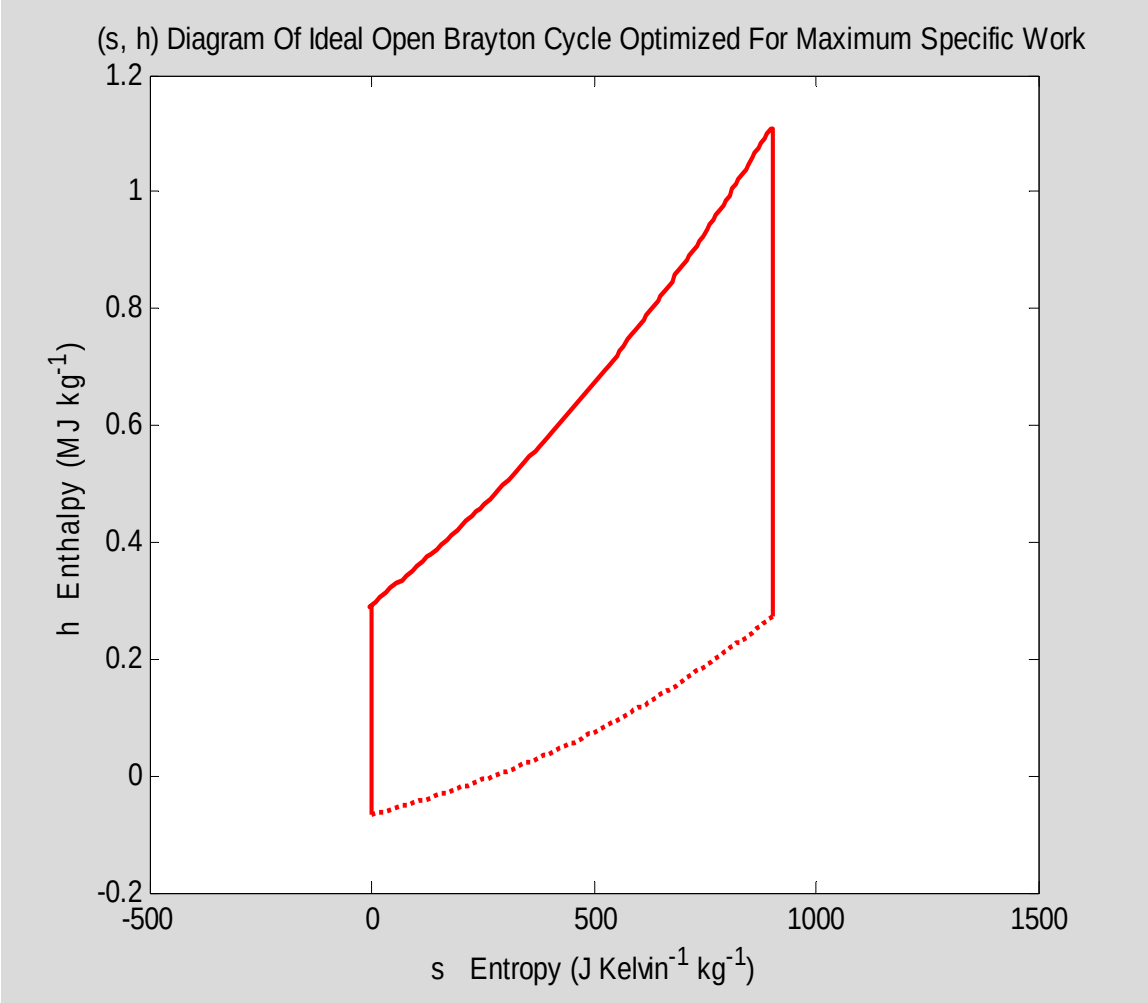


Figure 13-17: (s, h) Plane Ideal MarsMix Open Brayton Cycle Optimized For Maximum Specific Work

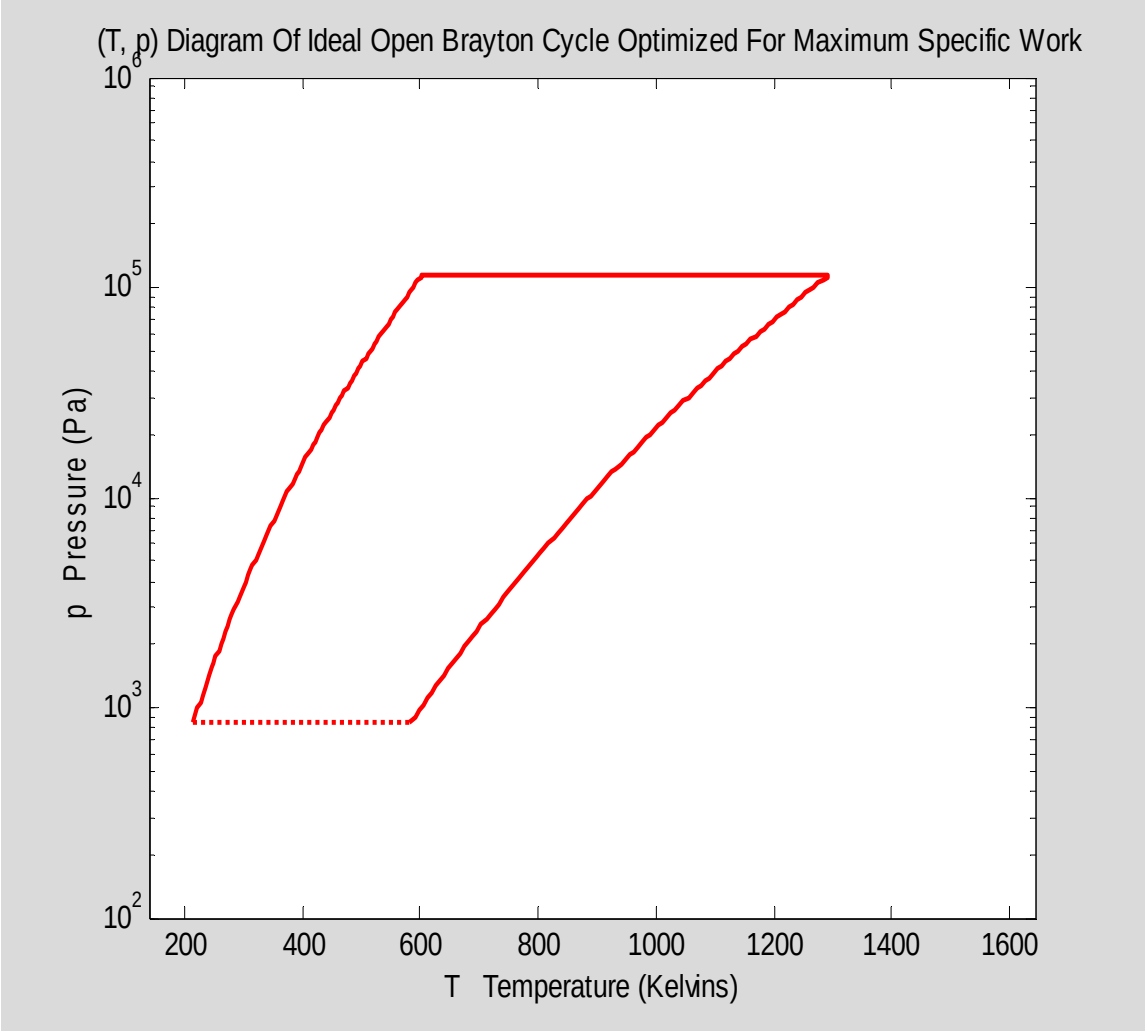


Figure 13-18: (T, p) Plane Ideal MarsMix Open Brayton Cycle Optimized For Maximum Specific Work

The adiabatic efficiency of a turbomachine can be directly determined from gas properties and the measured gas states of its intake and exhaust gas flows. For a compressor, the adiabatic efficiency is defined [Decher 1994, 197] as follows:

$$\eta_c \equiv \frac{\text{reversible work}}{\text{actual work for same pressure ratio}} \equiv$$
$$= \frac{h(T_{2,s}) - h(T_0)}{h(T_2) - h(T_0)} \tag{13-40a}$$

In Eq. 13-40, "h" refers to the gas enthalpy and the singly subscripted "T" symbols refer to *total* temperatures at the compressor's intake and exhaust, at locations as numbered in Figure 4-1. Note that the total temperature of a moving gas flow is defined as the temperature it would reach if its motion were adiabatically stopped. The symbol, $T_{2,s}$ refers to the temperature which the compressor's exhaust would have if the compressor had no losses but developed the same pressure ratio. For a gas with variable specific heats $T_{2,s}$ can be calculated using the table look-up methods of the previous section, i.e.,

$$T_{out} = h^{-1}\left[h[T_0] + \frac{E_c}{\eta_c} \right] \tag{13-40b}$$

so that the adiabatic efficiency formula becomes:

$$\eta_c^{(\text{constant } c_p)} = \frac{h\left[S_0^{-1}\left[S_0[T_0] + R\ln\left(\frac{p_2}{p_0}\right)\right]\right] - h[T_0]}{h[T_2] - h[T_0]} \tag{13-40c}$$

For a gas with constant specific heats it has a simpler algebraic form. This results in the following formula, which may be used to calculate the adiabatic efficiency of a compressor from operating measurements if the working gas has constant specific heats.

$$\eta_c^{(\text{constant } c_p)} = \frac{\left(\frac{p_2}{p_0}\right)^{\left(\frac{\kappa-1}{\kappa}\right)} - 1}{\left(\frac{T_2}{T_0}\right) - 1} \tag{13-41}$$

For an expander such as a turbine, the adiabatic efficiency is defined as follows:

$$\eta_E = \frac{\text{actual work}}{\text{reversible work for same pressure ratio}}$$
$$= \frac{h(T_4) - h(T_5)}{h(T_4) - h(T_{5,s})} \tag{13-42a}$$

where again, the numbering of Figure 4-1 applies and $T_{5,s}$ refers to the temperature which the turbine's exhaust would have if the turbine had the same pressure ratio but no losses. In the general case of variable specific heats this temperature can be calculated as:

$$h[T_{5,s}] = h\left[S_0^{-1}\left[S_0[T_4] - R\ln\left(\frac{p_4}{p_5}\right)\right]\right] \tag{13-42b}$$

so that the turbine expander's adiabatic efficiency is calculated by the following:

$$\eta_E^{(\text{constant } c_p)} = \frac{h[T_4] - h[T_5]}{h[T_4] - h\left[S_0^{-1}\left[S_0[T_4] - R\ln\left(\frac{p_4}{p_5}\right)\right]\right]} \tag{13-42c}$$

In the constant specific heat's case, Eq 13-42c can be restated as follows:

$$\eta_E^{(\text{constant } c_p)} = \frac{1 - \left(\frac{T_5}{T_4}\right)}{1 - \left(\frac{p_5}{p_4}\right)^{\left(\frac{\kappa-1}{\kappa}\right)}} \tag{13-43}$$

Unfortunately, component adiabatic efficiencies determined for one operating condition may not be valid for a different condition, even if the same turbomachine is being operated. The concept of polytropic efficiency was invented to address the need for a method of extending component efficiency information over a large range of possible operating conditions and even over classes of similar machines. The concept of polytropic efficiency is widely employed; every turbomachinery textbook consulted included a presentation of it.

However, none of the textbooks consulted included polytropic models for turbomachine operations using gases with variable specific heats. Since the martian air's specific heats are quite variable, the polytropic methods were extended to the variable specific heat case as part of this work.

In the adiabatic efficiency definitions, Eqs 13-40 and 13-42, the measured shaft power is not included in the definition. Instead, actual work and reversible work are inferred from measurements of the state of the gas alone. This has the effect of excluding from consideration any purely mechanical losses such as for instance friction within the rotating shaft's system of bearings. The losses that are included within the adiabatic efficiency definition are those that occur within the moving fluid stream. This has the convenient result for analyses that work losses remain in the gas. Even though work can be lost, the conservation of energy is still a valid principle. Any work losses in the moving fluid stream appear in the stream as heat.

For thermodynamic studies, the conceptual model of a lossy turbomachine is that it is equivalent to a cascaded series of small, identical, purely adiabatic and reversible compressor (or expander) components, interleaved with another cascaded series of components, i.e., small identical heat exchangers in which heat is added to the fluid. This cascaded small component concept is illustrated by the Figure 13-19 diagram.

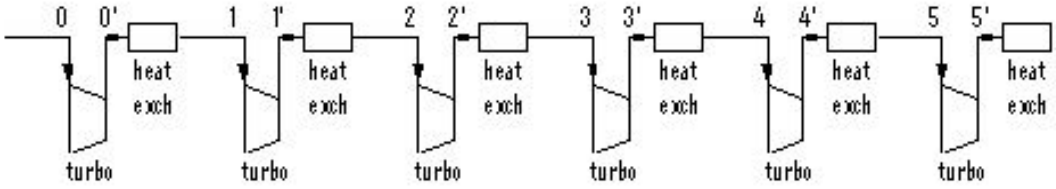


Figure 13-19: Conceptual Cascade Of Small Machines In Polytropic Models

The fluid's enthalpy change in one of the ideal compressor or expander stages matches the part of the work energy which is reversibly exchanged there with the fluid. The fluid's enthalpy change in the succeeding heat addition stage matches the remaining part of the work energy which is irreversibly converted to heat because of fluid friction losses.

The adiabatic process in the i^{th} compressor or expander stage is modeled as follows:

$$h_{i'} - h_i = -w_i^{(\text{reversible})} \tag{13-44}$$

$$s_{i'} - s_i = 0$$

Here, the conventional sign assignment practice for work is followed by defining w as the work done *by* the fluid. Thus, $(-w)$ is the work done *on* the fluid by an external agent.

The heat addition in the i^{th} heat exchanger stage which follows it in the conceptual cascade is modeled as isobaric:

$$h_{i+1} - h_{i'} = -w_i^{(\text{irreversible})} \tag{13-45}$$

$$p_{i+1} - p_{i'} = 0$$

Note that the sign convention used in Eq 13-45 also considers the irreversible work to be a negative number since it is an energy transfer *absorbed* by the fluid. It is not work being done by the fluid to an outside agent. Thus, $(-w_i^{(\text{irreversible})})$ is a *positive* numerical value for the energy dissipated as heat within the fluid.

Although these two processes are correctly stated in Eqs 13-44 and 13-45, they are represented in different thermodynamic planes. They must instead be stated in the same plane in order to combine them. Converting the second to the (s, h) plane by way of the lookup table function and inverse function mappings of Eqs 13-16, one obtains:

$$h_{i+1} - h_{i'} = -w_i^{(\text{irreversible})} \tag{13-46}$$

$$s_{i+1} - s_{i'} = S_0\left[h^{-1}\left[h_{i'} - w_i^{(\text{irreversible})}\right]\right] - S_0\left[h^{-1}\left[h_{i'}\right]\right]$$

Note that in the second of the Eq 13-16 equalities, the sign convention for work calls for subtraction of a *negative* irreversible work term, which is equivalent to adding a positive work value. Summing the corresponding Equations from 13-44 and 13-46 yields:

$$\Delta h_i \equiv h_{i+1} - h_i = \left(-w_i^{(\text{reversible})}\right) + \left(-w_i^{(\text{irreversible})}\right) \tag{13-47}$$
$$\Delta s_i \equiv s_{i+1} - s_i = S_0\left[h^{-1}\left[h_{i'} - w_i^{(\text{irreversible})}\right]\right] - S_0\left[h^{-1}\left[h_{i'}\right]\right]$$

Now, the sum of reversible and irreversible work done by the fluid is balanced by the total work coupled out from the turbomachine through its shaft, ignoring shaft friction. In a lossy compressor, the reversible work, the irreversible work, and the total work are all considered to be negative quantities because the work energy is transferred to the fluid instead of from the fluid. This balance statement for a compressor is therefore expressed by the following equation in which each quantity in parentheses is positive:

$$\left(-w_i^{(\text{reversible})}\right) + \left(-w_i^{(\text{irreversible})}\right) \equiv \left(-w_i^{(\text{total})}\right) \tag{13-48}$$

In a lossy expander, the reversible work and the total work are positive while the irreversible work is again negative. The total (positive) work done by the fluid is greater than the (positive) work coupled out through the rotating shaft because some of that fluid work is instead irreversibly dissipated as heat produced by flow friction losses. This balance statement for an expander such as a turbine is therefore expressed by the following equation, in which each quantity in parentheses is again positive:

$$\left(+w_i^{(\text{reversible})}\right) - \left(-w_i^{(\text{irreversible})}\right) \equiv \left(+w_i^{(\text{total})}\right) \tag{13-49}$$

Although Eqs. 13-48 and 13-49 are equivalent, they may appear different to some if positive numbers are to be used to characterize power ratings of both compressors and expanders. This difference, combined with the desire for efficiency values to only vary between zero and one, motivates the different efficiency definitions for compressors and expanders. Thus, for each compressor stage in the conceptual cascade the adiabatic efficiency as defined by Eq. 13-40 is the following ratio of positive quantities:

$$\eta_c = \frac{\left(-w_i^{(\text{reversible})}\right)}{\left(-w_i^{(\text{total})}\right)} \tag{13-50}$$

For each cascaded expander stage the adiabatic efficiency as defined by Eq. 13-42 is the following ratio of different positive quantities:

$$\eta_E = \frac{\left(+w_i^{(\text{total})}\right)}{\left(+w_i^{(\text{reversible})}\right)} \tag{13-51}$$

In the polytropic model, all of the conceptual cascaded stages are identical to each other. They have a single common ratio of reversible work to total work so their adiabatic efficiencies as per Eq. 13-50 or 13-51 are all the same throughout a cascade.

The polytropic model then goes to the infinite limit where an infinite sequence of infinitesimal adiabatic compressions or expansions is interleaved with an infinite sequence of infinitesimal heat additions, thus forming a continuous non-adiabatic compression or expansion process. The *polytropic* efficiencies, e_c and e_E, are defined as the limits of the single-stage adiabatic efficiencies as the conceptual cascade approaches an infinite number of stages. Indeed, in older texts the polytropic efficiency is also variously called the *small stage efficiency* or the *infinitesimal stage efficiency*. In this infinite limit, the total work per stage is replaced by a differential of the cumulative total fluid work done upstream of a particular location in the conceptual cascade. Invoking also Eqs13-50 and 13-51, Eqs 13-46 then become for a compressor (where $dw<0$):

$$\left\{\begin{array}{l} dh = -dw \\ ds = S_0\left[h^{-1}\left[h_{i'} - (1-e_c)dw\right]\right] - S_0\left[h^{-1}\left[h_{i'}\right]\right] \end{array}\right\}_{\text{compressor}} \tag{13-50}$$

and for an expander (where dw>0):

$$\left\{\begin{array}{l} dh = -dw \\ ds = S_0\left[h^{-1}\left[h_{i'} - \left(\frac{1-e_E}{e_E}\right)dw\right]\right] - S_0\left[h^{-1}\left[h_{i'}\right]\right] \end{array}\right\}_{\text{expander}} \tag{13-51}$$

Definitions of the enthalpy function, h, and the entropy integral function, S_0, involve integrals over temperature as stated in Equation 13-4b and 13-8. Substituting these definitions into the complicated expression above and algebraically simplifying it results in the following reduction:

$$\begin{aligned} & S_0\left[h^{-1}\left[h_{i'} + Cdw\right]\right] - S_0\left[h^{-1}\left[h_{i'}\right]\right] \equiv \\ & \equiv \frac{\left(\frac{dS_0[T]}{dT}\right)\Big|_{T=h^{-1}[h_{i'}]}}{\left(\frac{dh}{dT}\right)\Big|_{T=h^{-1}[h_{i'}]}} Cdw \equiv \frac{\left(\frac{c_p(T)}{T}\right)\Big|_{T=h^{-1}[h_{i'}]}}{\left(c_p(T)\right)\Big|_{T=h^{-1}[h_{i'}]}} Cdw \equiv \\ & \equiv \frac{C}{(T)\Big|_{T=h^{-1}[h_{i'}]}} dw \end{aligned} \tag{13-52}$$

Using the Eq. 13-52 reduction and eliminating the w variable by substitution, then Eqs 13-50 and 13-52 become for a compressor,

$$\left\{\begin{array}{l} ds=(1-e_c)\dfrac{dh}{T}\equiv(1-e_c)\dfrac{dh}{h^{-1}[h]} \\ \equiv(1-e_c)\dfrac{c_p(T)dT}{T}\equiv(1-e_c)dS_0[T] \\ \equiv(1-e_c)\dfrac{dh}{h^{-1}[h]} \end{array}\right\}_{\text{compressor}} \tag{13-53}$$

and for an expander:

$$\left\{\begin{array}{l} ds=\left(\dfrac{1-e_E}{e_E}\right)\dfrac{dh}{T}\equiv\left(\dfrac{1-e_E}{e_E}\right)\dfrac{dh}{h^{-1}[h]} \\ \equiv\left(\dfrac{1-e_E}{e_E}\right)\dfrac{c_p(T)dT}{T}\equiv\left(\dfrac{1-e_E}{e_E}\right)dS_0[T] \\ \equiv\left(\dfrac{1-e_E}{e_E}\right)\dfrac{dh}{h^{-1}[h]} \end{array}\right\}_{\text{expander}} \tag{13-54}$$

For the special case in which specific heats are constant, it is convenient to integrate the third forms given in Eqs 13-53 and 13-54 in order to obtain closed-form algebraic equations of the polytrope curves describing lossy turbomachines. Definitions of other properties allow these polytrope curves to be translated into other thermodynamic planes. The result is as follows for compressors using the MarsMix gas:

$$\left\{\begin{array}{l} s=s_0+(1-e_c)c_p\ln\left(\dfrac{T}{T_0}\right) \\ h=h_0+c_p(T-T_0) \\ \left(\dfrac{p}{p_0}\right)=\left(\dfrac{T}{T_0}\right)^{\left(e_c\frac{\kappa}{\kappa-1}\right)} \\ \left(\dfrac{\rho}{\rho_0}\right)=\left(\dfrac{T}{T_0}\right)^{\left(e_c\frac{\kappa}{\kappa-1}-1\right)} \end{array}\right\}^{(\text{constant } c_p)}_{\text{compressor polytrope}} \tag{13-55}$$

and for expanders such as turbines:

$$\left\{\begin{array}{l} s = s_0 + \left(\dfrac{1-e_E}{e_E}\right) c_p \ln\left(\dfrac{T}{T_0}\right) \\ h = h_0 + c_p\left(T - T_0\right) \\ \left(\dfrac{p}{p_0}\right) = \left(\dfrac{T}{T_0}\right)^{\left(\frac{\kappa}{e_E(\kappa-1)}\right)} \\ \left(\dfrac{\rho}{\rho_0}\right) = \left(\dfrac{T}{T_0}\right)^{\left(\frac{\kappa}{e_E(\kappa-1)}-1\right)} \end{array}\right\}_{\text{expander polytrope}}^{(\text{constant } c_p)} \tag{13-56}$$

Clearly there is a direct mathematical relation based on their definitions alone between adiabatic efficiency and polytropic efficiency. For gases with constant specific heats, it is expressed as follows [Decher 1994, 199]:

$$\eta_c^{(\text{constant } c_p)} = \frac{\left(\dfrac{p_2}{p_0}\right)^{\left(\frac{\kappa-1}{\kappa}\right)} - 1}{\left(\dfrac{p_2}{p_0}\right)^{\left(\frac{\kappa-1}{\kappa e_c}\right)} - 1} \tag{13-57a}$$

and its inverse relation is:

$$e_c = \frac{\left(\dfrac{\kappa-1}{\kappa}\right)\ln\left(\dfrac{p_2}{p_0}\right)}{\ln\left(1 + \dfrac{1}{\eta_c^{(\text{constant } c_p)}}\left(\left(\dfrac{p_2}{p_0}\right)^{\left(\frac{\kappa-1}{\kappa}\right)} - 1\right)\right)} \tag{13-57b}$$

In these equations, e_c is the polytropic efficiency of the compressor and η_c is its adiabatic efficiency. For an expander such as a turbine, the corresponding relations are as follows:

$$\eta_E^{(\text{constant } c_p)} = \frac{\left(\dfrac{p_4}{p_5}\right)^{\left(e_E\frac{\kappa-1}{\kappa}\right)} - 1}{\left(\dfrac{p_4}{p_5}\right)^{\left(\frac{\kappa-1}{\kappa}\right)} - 1} \tag{13-58a}$$

and the inverse is:

$$e_E = \frac{\ln\left(1+\left(\left(\frac{p_4}{p_5}\right)^{\left(\frac{\kappa-1}{\kappa}\right)}-1\right)\eta_E^{\,(\text{constant } c_p)}\right)}{\left(\frac{\kappa-1}{\kappa}\right)\ln\left(\frac{p_4}{p_5}\right)} \qquad (13\text{-}58b)$$

The primary reason that the polytropic model is important is the claim, reiterated in all texts surveyed, that the polytropic efficiency of a turbomachine remains approximately constant over a wide range of operating conditions while the adiabatic efficiency does not. Indeed, the claim is also made that classes of similar turbomachines have almost the same polytropic efficiencies across their class. Thus, a particular assumed numerical value of polytropic efficiency determines approximately how the associated adiabatic efficiency varies over the entire space of possible operating conditions.

A second reason that the polytropic model is important is the convenience it affords for gases with constant specific heats. For these gases it is possible to determine an associated polytropic exponent, usually denoted by the symbol, n, such that the equation of an adiabat curve in the (V_{sp}, p) thermodynamic plane,

$$pV_{sp}^{\ \kappa} = \text{constant}$$

is replaced by the equation of a polytrope curve,

$$pV_{sp}^{\ n} = \text{constant}$$

By using the symbol n instead of κ in some of the formulas derived for ideally efficient processes, the calculated results correctly reflect polytropic loss cases. Thus, the polytropic turbomachine model simplifies cycle calculations analyzing performance with lossy components, at least for thermal conversion gases with constant specific heats.

However, for the present nuclear engine design, the variable specific heats of martian air dictate not using the formulas in Eqs. 13-55 through 13-58. Instead, polytropic algorithms are needed for variable specific heat gases such as the MarsMix gas which represents the martian air. Such algorithms have been derivd as part of the present work. The fourth equivalent form appearing in Eqs 13-53 and 13-54 are perfect differential forms which therefore can be integrated directly. Thus, for the general polytropic loss case in which the thermal conversion working gas may have variable specific heats, we have the following relating compressor output and intake states, i.e., states 2 and 0 of Figure 4-1 :

$$\left\{\begin{array}{l} s_2 = s_0 - (1-e_c)\left(S_0[T_2] - S_0[T_0]\right) \\ h_2 = h[T_2] = h[T_0] + (-w_c) \\ p_2 = p_0 \exp\left(e_c \dfrac{S_0[T_2] - S_0[T_0]}{R} \right) \end{array}\right\}_{\text{compressor}} \tag{13-59}$$

Similarly, for the turbine expander we have the following relating its exhaust and intake states, i.e., states 5 and 4 of Figure 4-1:

$$\left\{\begin{array}{l} s_5 = s_4 - \left(\dfrac{1-e_E}{e_E} \right)\left(S_0[T_4] - S_0[T_5]\right) \\ h_5 = h[T_5] = h[T_4] - w_E \\ p_5 = p_4 \exp\left(-\dfrac{1}{e_E} \dfrac{S_0[T_4] - S_0[T_5]}{R} \right) \end{array}\right\}_{\text{expander}} \tag{13-60}$$

Equations 13-59 and 13-60 define polytrope curves for lossy compressors and expanders operating with a gas which may have variable specific heats. They are less convenient to use than the closed-form algebraic equations which apply to constant specific heat cases, but they can be numerically applied to cycle analyses provided that look-up interpolation tables are available for the entropy integral function $S_0[T]$ and the enthalpy function $h[T]$. Since these tables have been implemented for MarsMix gas in MATLAB code (see Appendix C), Eqs 13-59 and 13-60 are used herein.

Combining Eqs 13-59 and 13-60, for the general case where the working gas has variable specific heats, with Eqs 13-40c and 13-42c, leads to the following general formulae to convert from polytropic efficiencies to adiabatic efficiencies:

$$\eta_c = \frac{h\left[S_0^{-1}\left[S_0\left[T_0 + R \ln \dfrac{p_2}{p_0} \right] \right] \right] - h[T_0]}{h\left[S_0^{-1}\left[S_0\left[T_0 + \dfrac{R}{e_c} \ln \dfrac{p_2}{p_0} \right] \right] \right] - h[T_0]} \tag{13-61a}$$

$$\eta_E = \frac{h[T_4] - h\left[S_0^{-1}\left[S_0\left[T_4 - e_E R \ln \dfrac{p_4}{p_5} \right] \right] \right]}{h[T_4] - h\left[S_0^{-1}\left[S_0\left[T_4 - R \ln \dfrac{p_4}{p_5} \right] \right] \right]} \tag{13-61b}$$

These can be inverted using straightforward but tedious algebraic steps to give the following complicated general formula to convert in the reverse direction, i.e., from adiabatic efficiencies to polytropic efficiencies

$$e_c = \frac{R \ln \frac{p_2}{p_0}}{S_0\left[h^{-1}\left[\frac{h\left[S_0^{-1}\left[S_0[T_0] + R \ln \frac{p_2}{p_0} \right] \right] - h[T_0]}{\eta_c} + h[T_0] \right] \right] - S_0[T_0]} \eta_c \qquad (13\text{-}62a)$$

$$e_E = \frac{S_0[T_4] - S_0\left[h^{-1}\left[h[T_4] - \eta_E \left(h[T_4] - h\left[S_0^{-1}\left[S_0\left[T_4 - R \ln \frac{p_4}{p_5} \right] \right] \right] \right) \right] \right]}{R \ln \frac{p_4}{p_5}} \qquad (13\text{-}62b)$$

Polytrope curves, calculated via Eq. 13-59 for compressors of different polytropic efficiencies compressing the MarsMix gas representing martian air are presented in Figures 13-20, 13-21, and 13-22 which are plotted respectively in (s, T), (s, h), and (T, p) thermodynamic planes. Their different curves reflect polytropic efficiencies ranging from 100% down to 40% at 10% steps, with the 100 % polytropic efficiency curve (which is an adiabat) on the upper left. Spacings between curve markers represent equal increments of total external work (per unit fluid mass flow) transferred between the shaft and the fluid. The entropy zero reference point is arbitrary.

Polytropic curves similarly calculated via Eq. (13-60) for turbine expanders using MarsMix gas appear in the Figures 13-23, 13-24, and 13-25.. The 100% efficiency curves (which are adiabats) are on the lower right.

13.7.3 Heat Exchanger Pressure Losses

From the perspective of the gas, a heat exchanger for the present nuclear engine design appears simply as a duct with heated walls into which the compressed gas flows as it emerges from the compressor. Its purpose is to add heat from the nuclear reactor to the gas before the gas flows into the turbine. However, friction between the flowing gas and the duct's walls also causes an unwanted pressure drop which reduces cycle efficiency.

The full scenario of duct flow of a perfect gas with constant specific heats in a frictional constant-area duct with heat transfer through its walls is analyzed in M. Saad's (13-21a)

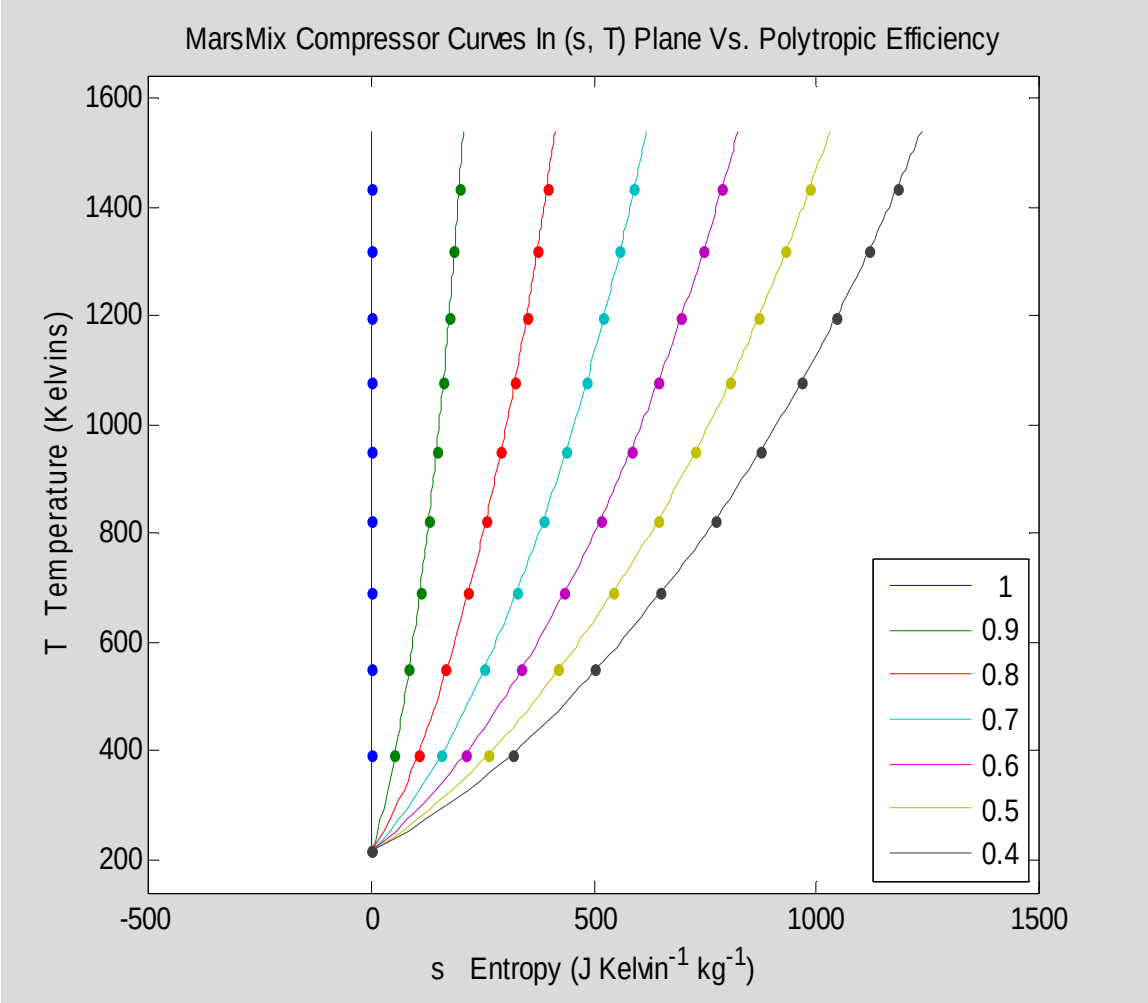


Figure 13-20: (s, T) Plane Plot Of MarsMix Compressor Performance Vs Polytropic Efficiency

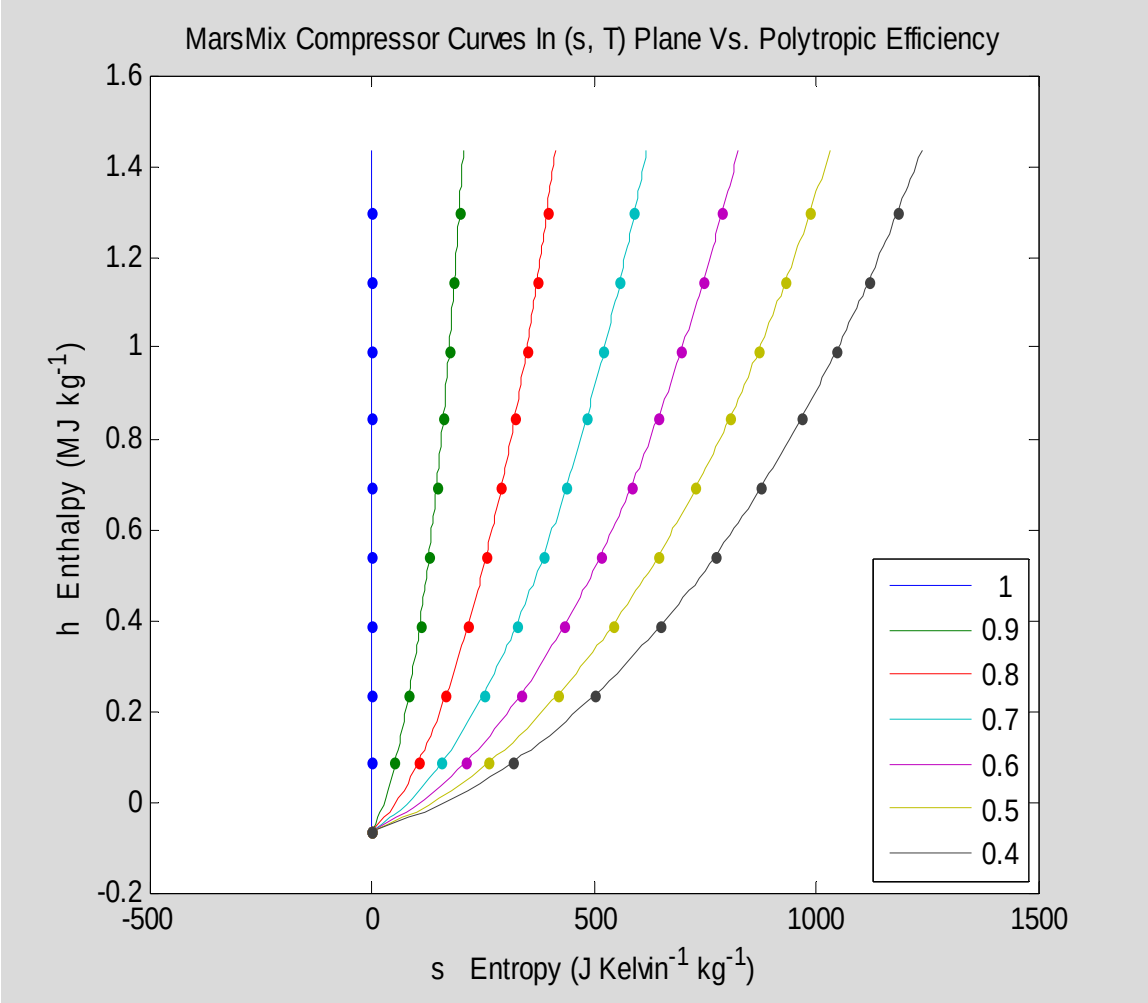


Figure 13-21: (s, h) Plane Plot Of MarsMix Compressor Performance Vs Polytropic Efficiency

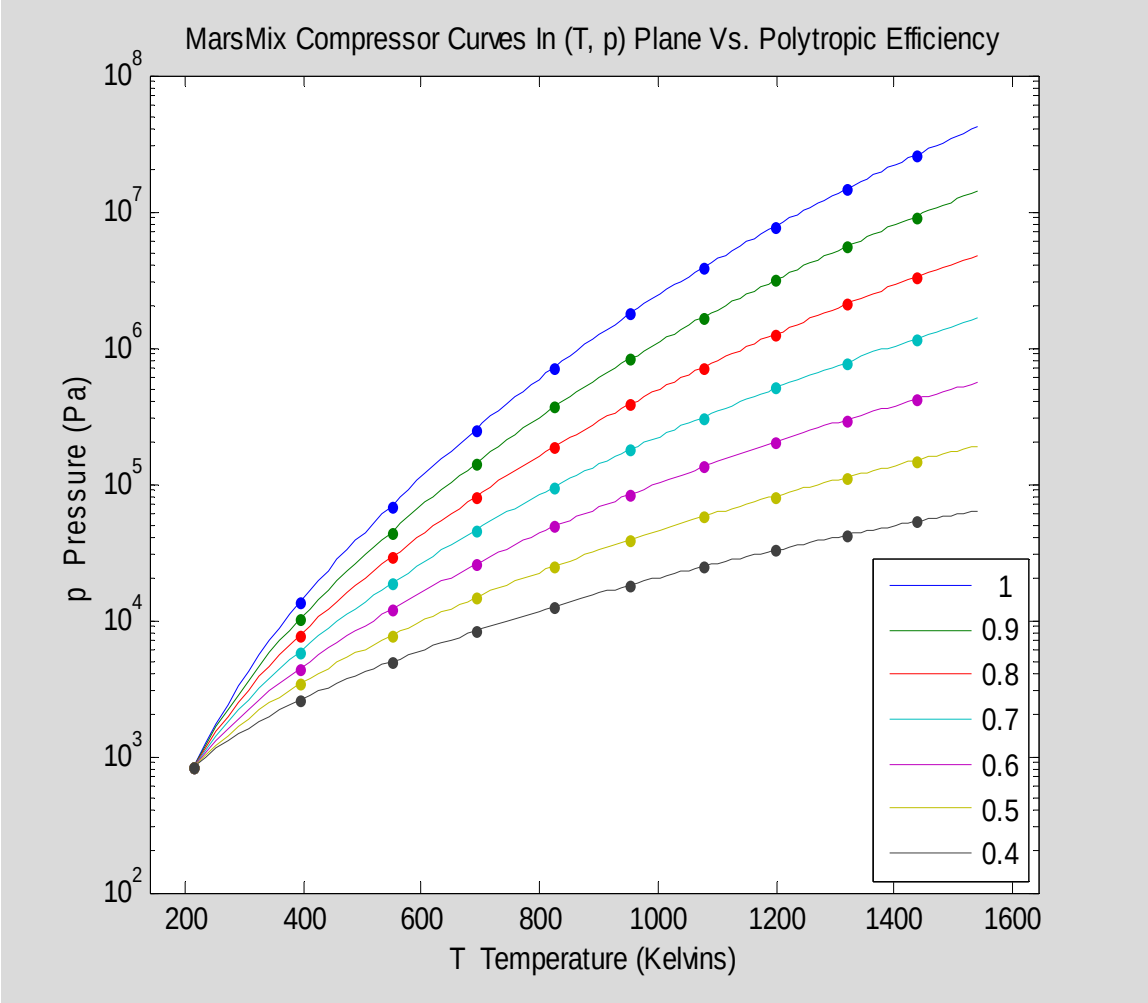


Figure 13-22: (T, p) Plane Plot Of MarsMix Compressor Performance Vs Polytropic Efficiency

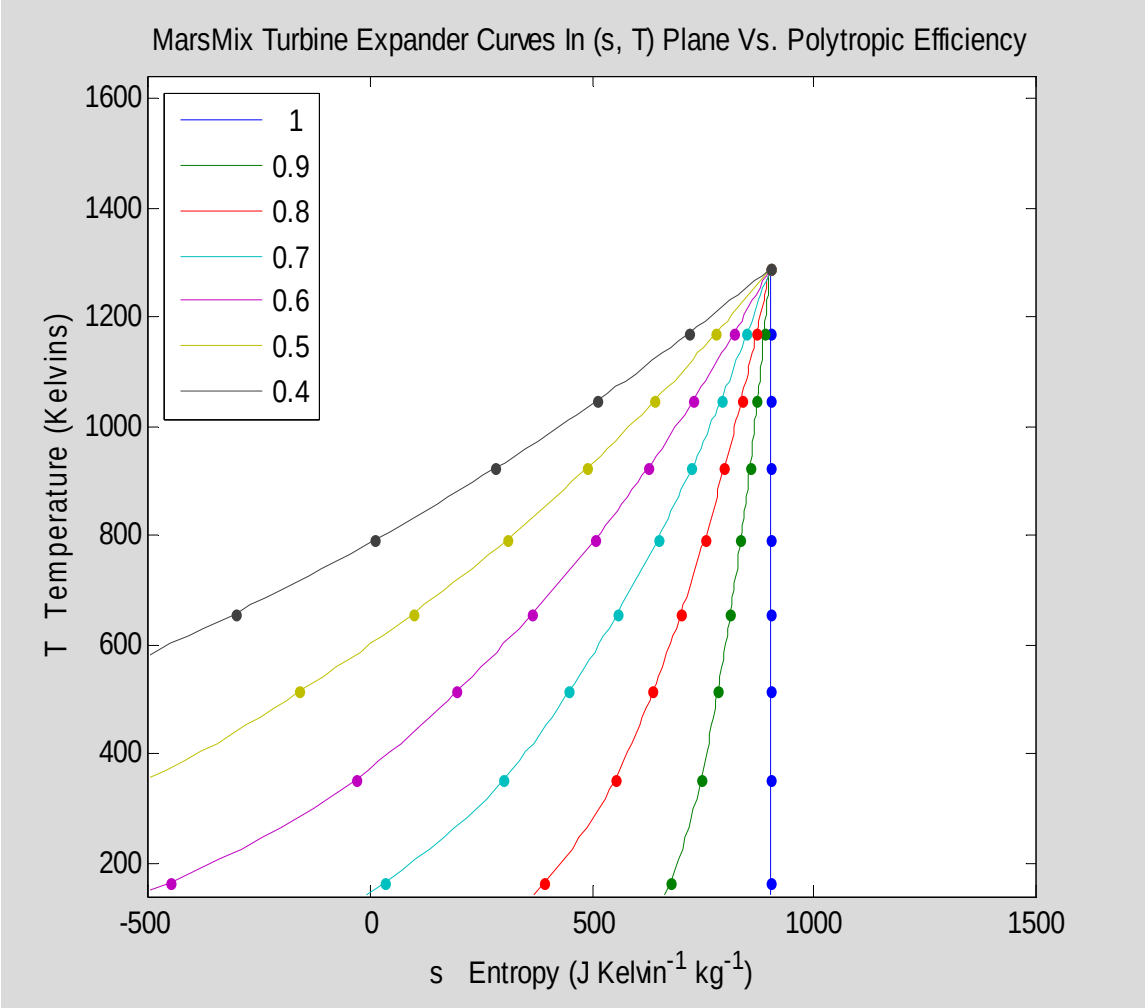


Figure 13-23: (s,T) Plot Of MarsMix Turbine Expander Performance Vs Polytropic Efficiency

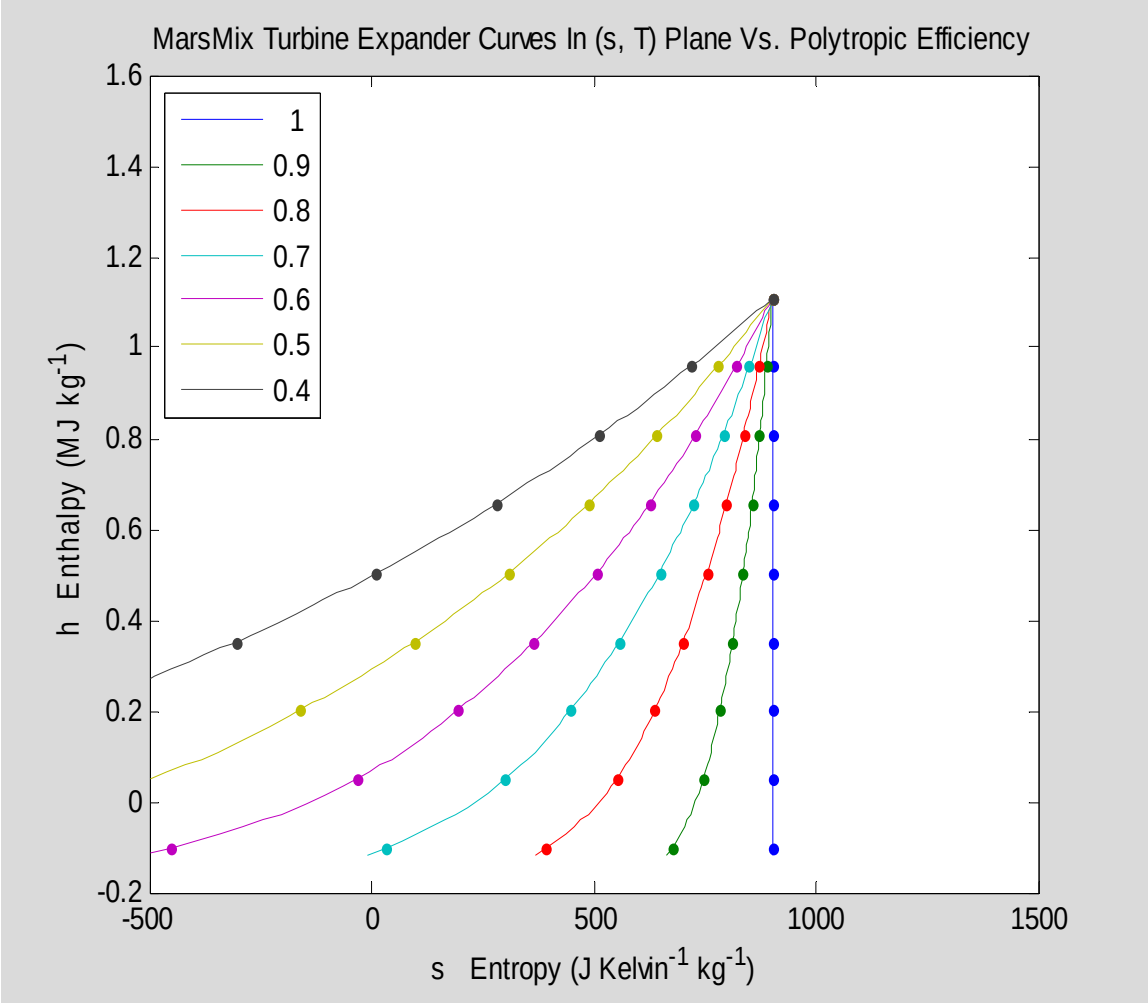


Figure 13-24: (s,h) Plot Of MarsMix Turbine Expander Performance Vs Polytropic Efficiency

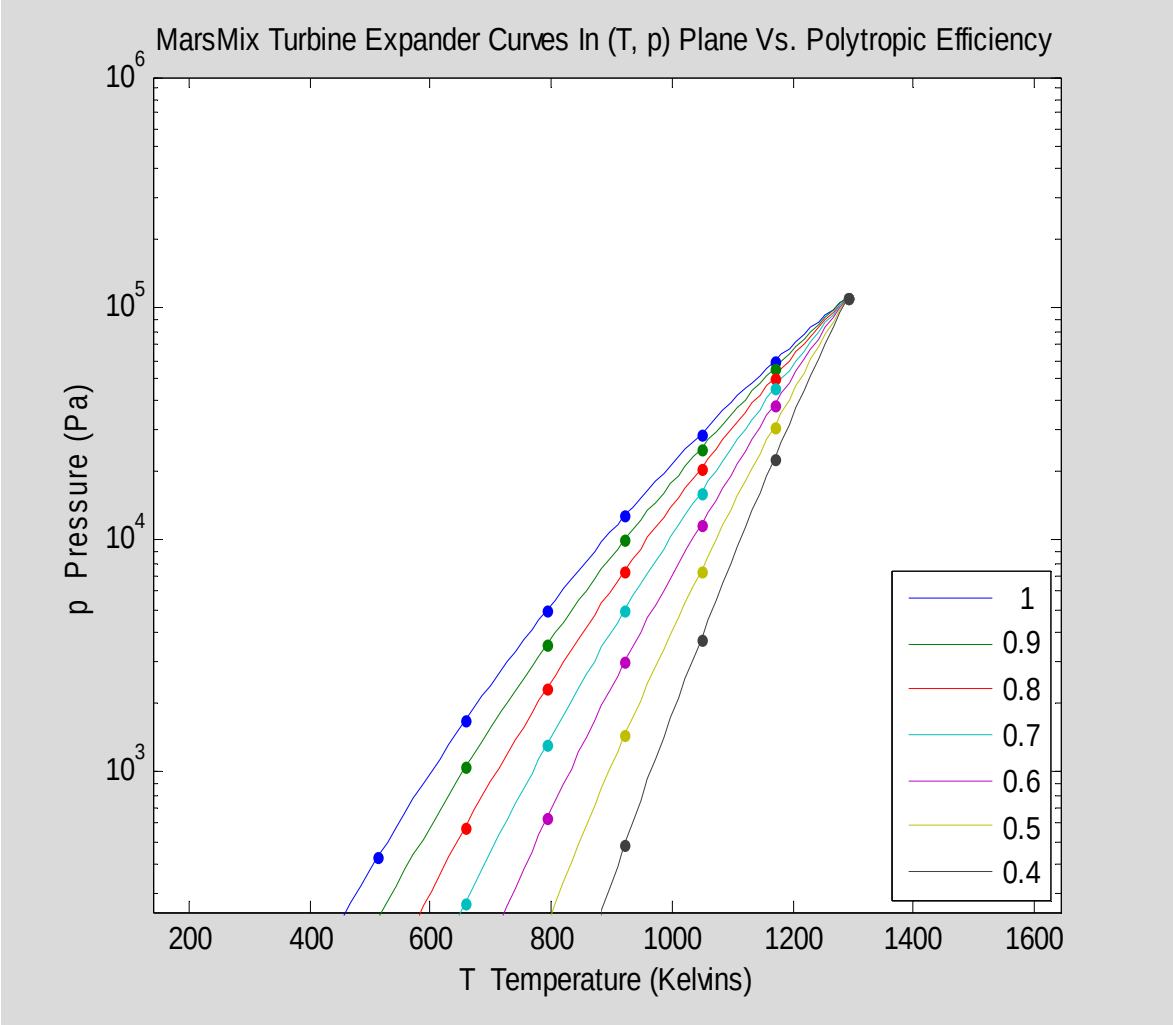


Figure 13-25: (T, p) Plot Of MarsMix Turbine Expander Performance Vs Polytropic Efficiency

Compressible Fluid Flow textbook (Saad 1985, 190-204, 275-288). The resulting system of coupled ordinary differential equations defining fluid properties as a function of 1-D position in the duct is quite complicated, involving nonlinear functions of the flow's Mach number in addition to the static thermodynamic gas properties.

The gas flow situation is complicated even if no heat through walls is added. For subsonic flows entering a duct, the pressure decrease due to internal friction as the gas flows downstream cause the specific volume of the gas to increase. Since the duct has constant cross-sectional area, the gas velocity increases as it travels. The kinetic energy paying for this gas velocity increase is naturally removed from the gas enthalpy. Thus, subsonic gas traveling in an unheated duct accelerates and cools as it travels; its entropy increases due to the friction while its enthalpy decreases. The situation is partially reversed if the gas flow is supersonic in which case the gas decelerates as it travels in the duct with its temperature and enthalpy now increasing, although its entropy still increases due to the friction. The zero-heat-added situation leads to the famous Fanno line in the (s, h) plane, a nonlinear curve describing the thermodynamic trajectories of gas flows of all speeds through unheated ducts with friction. One of the expressions derived in Saad's book [Saad 1985, 198] gives the slope of the Fanno line curve as:

$$\frac{dh}{ds} = \frac{\kappa T M^2}{M^2 - 1} \tag{13-63}$$

where M represents the local Mach number of the gas flow at a point in a duct.

With heat transfer through the duct walls also adding heat to the gas, the acceleration of the gas is increased which causes further changes to the Mach number and the flow friction profiles, further complicating the situation. Enthalpy and entropy changes due to heat addition are added to the (s, h) trajectory direction of the zero-heat-added case.

In order to simplify heat exzchanger analyses in this chapter, it is assumed that gas speeds within the heat exchanger are much smaller than the speed of sound, i.e., that $M<<1$. Then M^2 becomes negligible so the derivative in Eq. 13-63 is approximated as zero. Therefore, the end-to-end change in gas enthalpy within a heat exchanger is modeled here as exactly matching the heat addition. Thus, Eq. 13-21a (repeated here) still applies in this approximation:

$$\left(h[T_4] - h[T_2]\right)\dot{m} = \dot{q}$$

Also to simplify analyses, the end-to-end pressure drop in the heat exchanger is specified as an input variable from which other quantities are calculated instead of itself being calculated from other flow variables:

$$\Delta p_{heatex} = p_2 - p_4 \tag{13-64a}$$

In the present calculations this heat exchanger pressure drop is set through a specified per-unit value unconventionally defined here to serve as a kind of pressure efficiency:

$$e_p \equiv \frac{p_4}{p_2} = 1 - \left(\frac{\Delta p_{heatex}}{p_2} \right) \tag{13-64b}$$

so that

$$\Delta p_{heatex} \equiv p_2 \left(1 - e_p\right) \tag{13-64c}$$

The end-to-end effect of the heat exchanger on entropy is then:

$$s_4 = s_2 + S_0[T_4] - S_0[T_2] - R \ln\left(\frac{p_4}{p_2} \right) \tag{13-65}$$

For the purposes of identifying a trajectory in the (s, h) plane to represent the internal changes within the heat exchanger, the enthalpy and pressure quantities are interpolated. Thus, if ξ is a dummy independent variable ranging from 0 to 1 within the heat exchanger, then the trajectory is defined parametrically as follows:

$$p_\xi = p_2 - (\Delta p_{heatex})\xi$$
$$h_\xi = h_2 + (h_4 - h_2)\xi$$
$$T_\xi = h^{-1}[h_\xi]$$
$$s_\xi = s_2 + S_0[T_\xi] - S_0[T_2] - R \ln\left(\frac{p_\xi}{p_2} \right) \tag{13-66}$$

13.7.4 Open Brayton Cycle Performance With Lossy Components

Just as with the ideal cycle, the lossy Open Brayton Cycle is fully characterized by its gas conditions at the locations which in Figure 4-1 are numbered 0, 2, 4, and 5. In particular, Eq. 13-21 (repeated here for convenience) fully applies.

$$(h[T_4] - h[T_2])\dot{m} = \dot{q}$$
$$(h[T_0] - h[T_2] + h[T_4] - h[T_5])\dot{m} = P_{\text{out}} \tag{13-21(a, b))}$$

It is still true in the lossy case that the gas conditions at the air intake location including air temperature, T_0, and pressure, p_0, are taken as given. It's still true that the maximum gas temperature is determined by constraints (since if constraints did not limit it then it would be beneficial to increase it until they did). It's still true that the compressor's pressure ratio is the fundamental parameter to be chosen by the designer. The situation differs from the ideal case only in the interrelations between gas states, now described by Eqs 13-59, 13-60, 13-64, and 13-65. With some algebraic manipulations, these can be assembled into a computational scheme in which, for any value of the compressor's

specific work, the remaining variables are fully determined based on assumed efficiencies and limiting temperatures.

This computational scheme, which was implemented in MATLAB (see Appendix F) determines the efficiency vs. compression pressure ratio performance measure function and the specific work vs compression pressure ratio performance measure function of an Open Brayton Cycle (OBC) with lossy components, using the MarsMix gas simulating martian air. The algorithm only needs to be given a triplet of the loss parameter values, e_c, e_E, and e_p to generate its two curves for fixed temperature limits, T_0 and T_4. . Although the fundamental design parameter is the compressor's pressure ratio, π_c, this algorithm instead starts with assumed values of total compressor work per unit mass, i.e., $(-w_c)$. The corresponding pressure ratios are then calculated.

In detail, the computational algorithm for evaluating open Brayton cycle performance in Martian air using lossy components is as follows:

- Choose the martian air intake pressure and temperature, p_0 and T_0 (e.g., 850 Pa and 215 Kelvins)
- Choose the maximum temperature of the heated compressed martian air, T_4 (e.g., 1290 Kelvins).
- Pick any positive value for the compressor work per unit mass flow, $(-w_c)$
- Calculate intake enthalpy via interpolation tables, $h_0 = h[T_0]$
- Calculate maximum enthalpy from interpolation tables, $h_4 = h[T_4]$
- Calculate compressor outlet enthalpy, $h_2 = h_0 + (-w_c)$
- Calculate compressor outlet temperature by inverse interpolation, $T_2 = h^{-1}[h_3]$
- Calculate compressor pressure ratio from interpolated entropy integral function, $\pi_c = \exp\left(e_c \frac{S_0[T_2] - S_0[T_0]}{R_{MarsMix}} \right)$
- Calculate cycle pressures, $p_2 = \pi_c p_0$ and $p_4 = e_p p_2$
- Calculate exhaust temperature using both inverse and forward interpolation of the entropy integral function, $T_5 = S_0^{-1}\left[S_0[T_4] - e_E R_{MarsMix} \ln\left(\frac{p_4}{p_0} \right) \right]$
- Calculate exhaust enthalpy via interpolation table, $h_5 = h[T_5]$
- Calculate turboexpander work per unit mass flow, $w_E = h_4 - h_5$
- Calculate **specific work** (i.e., net work per unit mass flow), $w_{sp} = w_E - (-w_c)$
- Calculate heat addition per unit mass flow, $(\dot{q}/\dot{m}) = h_4 - h_5$
- Calculate energy conversion cycle **efficiency**, $\eta = \frac{w_{sp}}{(\dot{q}/\dot{m})}$

As long as $(-w_c)$ is not too large the algorithm's calculated compressor outlet temperature, T_2, will remain properly less than T_4. Conditions violating this limit must be rejected.

This computational algorithm evaluating lossy OBC implementations in martian air was run for many cases to generate a data base relating the cycle performance measures, i.e., specific work and efficiency, to compressor pressure ratio, temperatures, compressor work, and polytropic efficiencies. Plots were generated from the resulting data base.

Instead of separately plotting the resulting performance measures versus the compression pressure ratio, it is informative to instead cross plot them against each other with the compressor pressure ratio suppressed, as was done in Figures 13-13 and 13-14. Figures 13-26 through 13-29 show families of such specific work vs efficiency cross-plot curves, with the heat exchanger pressure ratio set at 100% in Figure 13-26, at 80% in Figure 13-27, at 60% in Figure 13-28, and at 40% in Figure 13-29. The separate curves within each figure represent different choice of turbocompressor and turboexpander losses, but only curves representing identical turbocompressor and turboexpander polytropic efficiencies are plotted in these graphs. The eight polytropic efficiency values evaluated and plotted range from 65 % to 100 % in increments of 5 %.

A review of the plots reveals that performance is much more sensitive to turbomachine inefficiencies than it is to the heat exchanger's pressure drop. Reducing turbomachine efficiency from 100% to 90% is similar in its effect on overall OBC performance to reducing the heat exchanger pressure fraction from 100% to 40%.

Another observation is that the shape of the cross-plot curves as evident in Figures 13-13 and 13-14 are only found here in the single curve of Figure 13-26 in which the three components are all ideally perfect, i.e., with no losses. As soon as any small inefficiency is introduced in any of the modeled components, both ends of the resulting cross-plot curve terminate at the origin. Therefore, for practical combinations of the component loss parameters, the highest possible cycle efficiency is not much higher than the efficiency that would result if the maximum specific work were chosen as the design point. Study of the curves reveals further that as turbomachine efficiencies are reduced into realistic ranges the discrepancy between the maximum specific work design point and the maximum efficiency design point becomes smaller.

It may be useful to tentatively decide to adopt the maximum specific work design point's compressor pressure ratio, thus eliminating the cross-plot curves along with all their complexity from consideration and replacing each curve with a single design point. It would then be possible to investigate how that optimized design's particular specific work and efficiency values change as a function of component loss parameters.

This has been done in another MATLAB calculation. With cycle temperature limits held fixed at 215 and 1290 Kelvins and with the heat exchanger pressure ratio held fixed at 100%, the resulting plots show how for optimized maximum specific work designs, the actual values of the optimized compressor pressure ratio, of the optimized cycle efficiency and the optimized specific work all vary jointly as functions of the polytropic efficiencies of the compressor and turbine.

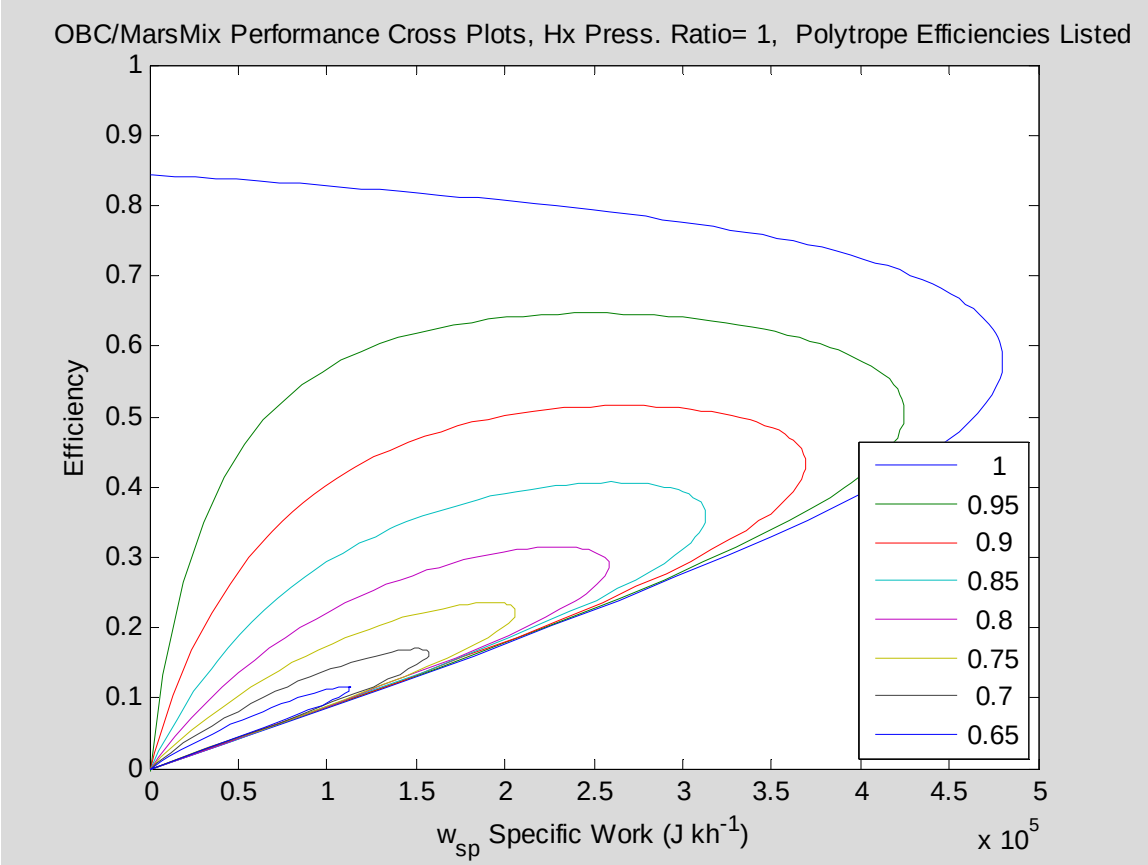


Figure 13-26: Specific Work Vs Cycle Efficiency Cross Plots For A Range Of Turbomachine Polytropic Efficiencies And For A Heat Exchanger Pressure Ratio Of 1.00

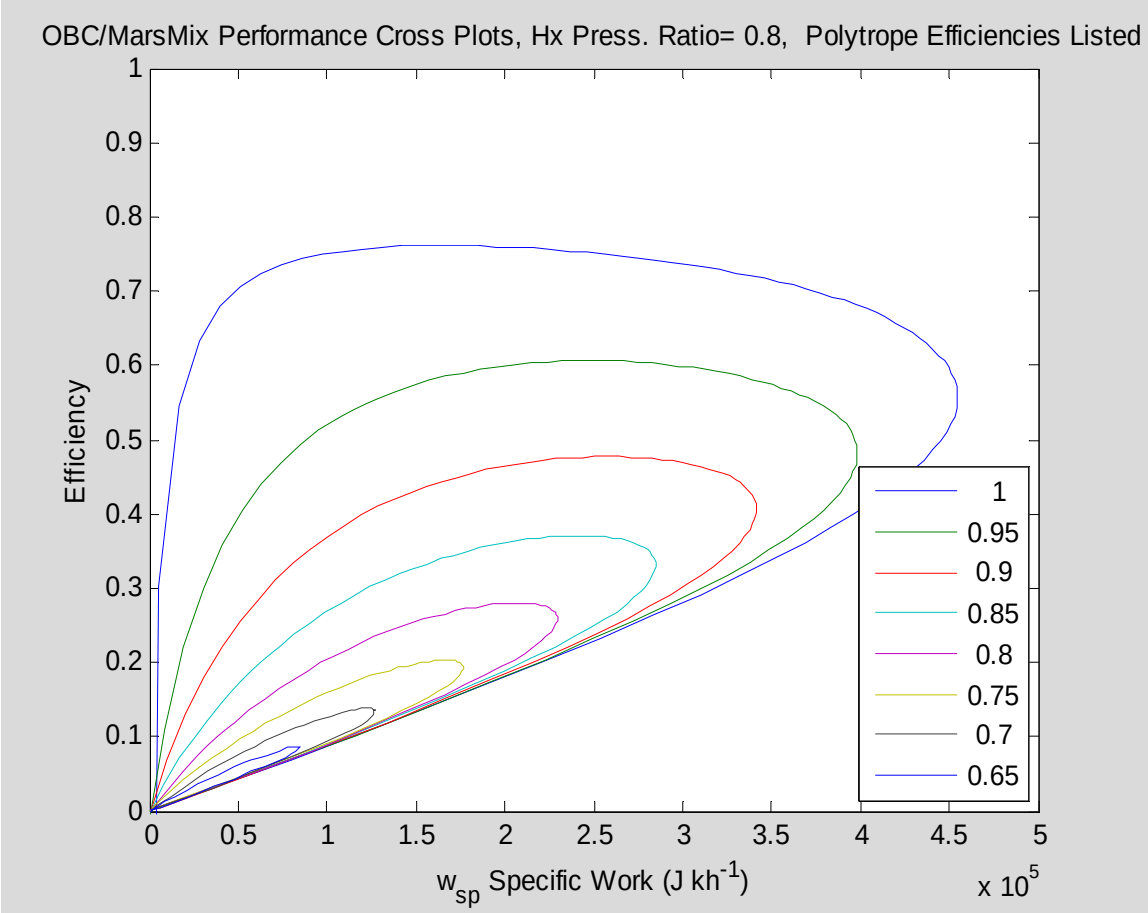


Figure 13-27: Specific Work Vs Cycle Efficiency Cross Plots For A Range Of Turbomachine Polytropic Efficiencies And For A Heat Exchanger Pressure Ratio Of 0.80

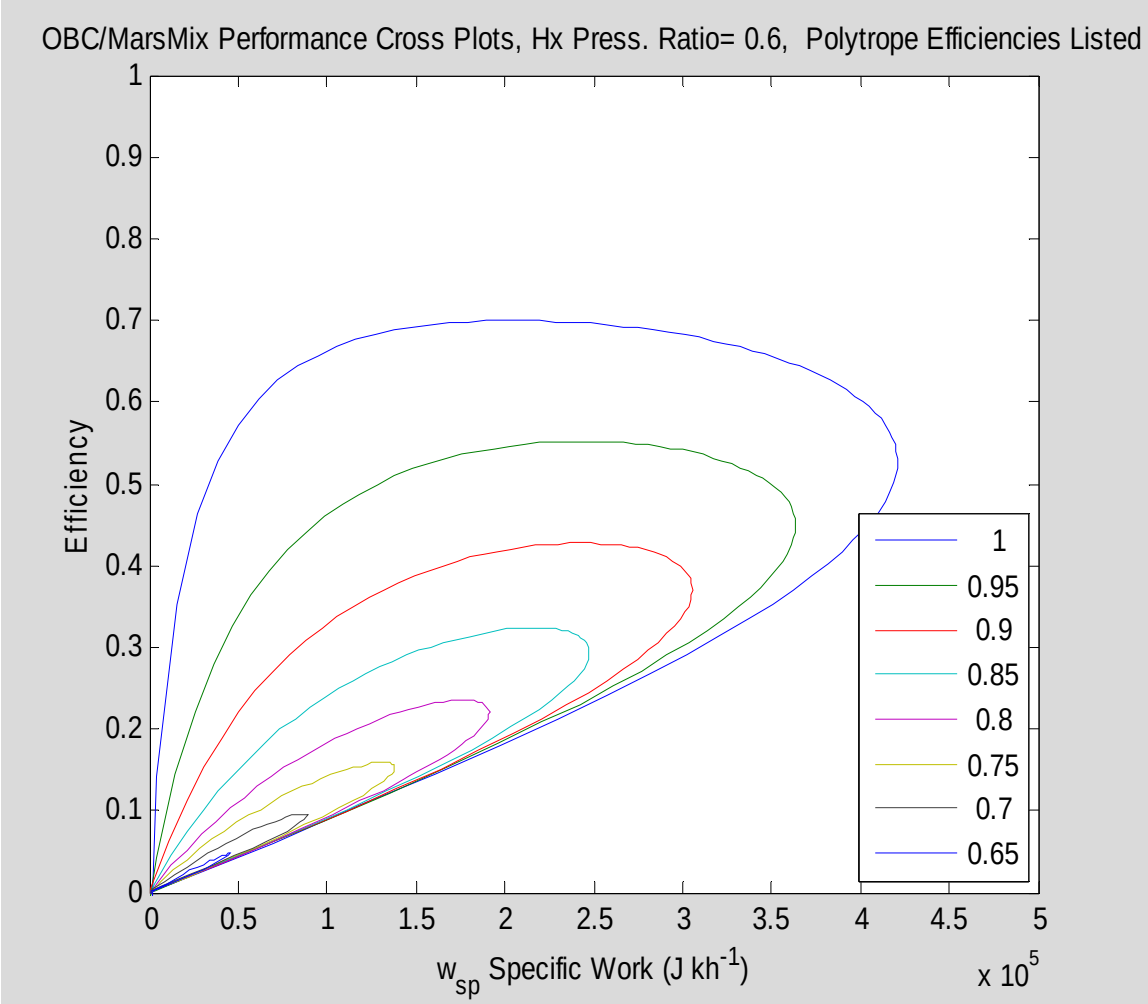


Figure 13-28: Specific Work Vs Cycle Efficiency Cross Plots For A Range Of Turbomachine Polytropic Efficiencies And For A Heat Exchanger Pressure Ratio Of 0.60

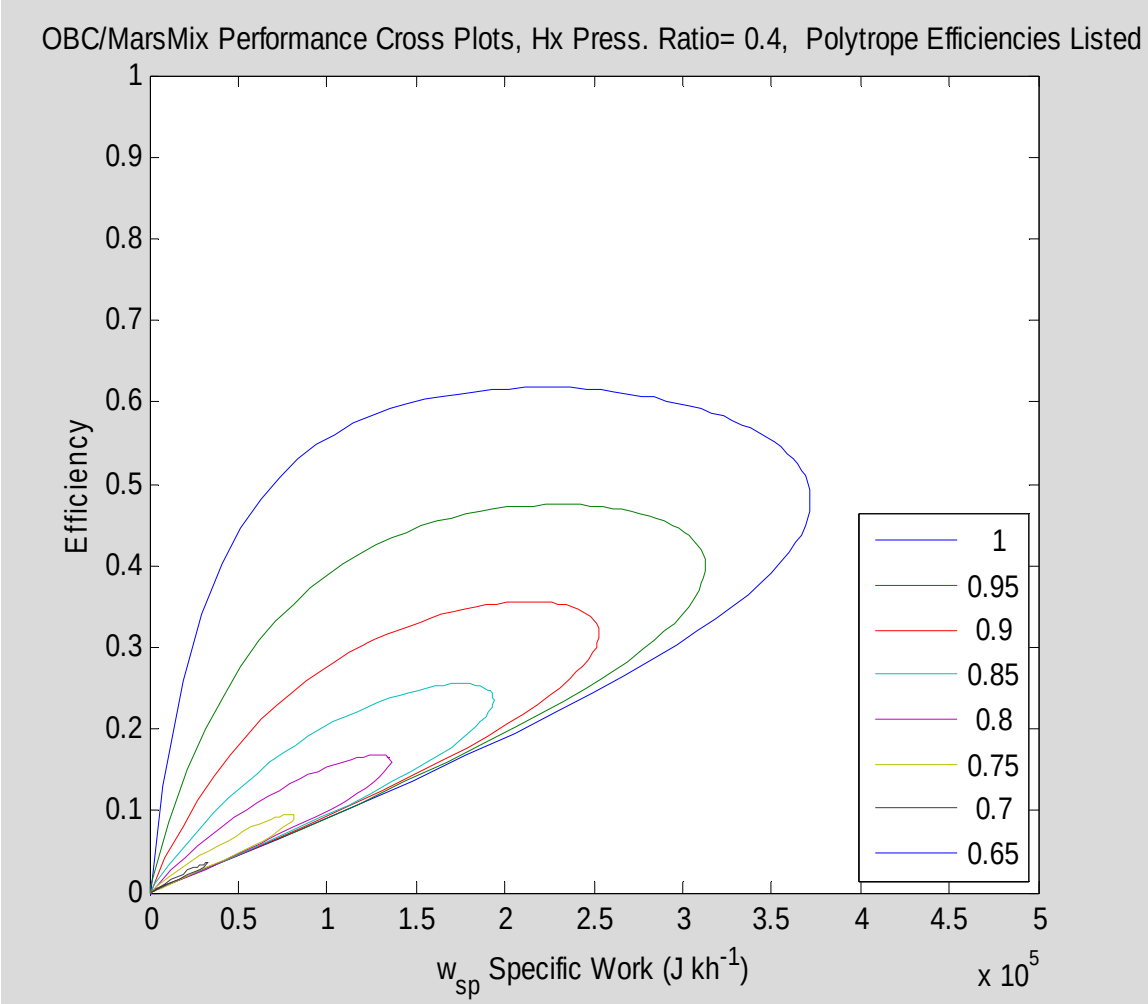


Figure 13-29: Specific Work Vs Cycle Efficiency Cross Plots For A Range Of Turbomachine Polytropic Efficiencies And For A Heat Exchanger Pressure Ratio Of 0.40

For an ideal lossless heat exchanger these plots appear as Figures 13-30 through 13-32. For all three plots the axes are the polytropic efficiencies of the compressor and turbine respectively. In Figure 13-30, contour plots against these axes present the compressor pressure ratio of designs optimized for maximum specific work. Figure 13-31 contour plots the cycle efficiency of designs optimized for maximum specific work. Figure 13-32 contour plots the the specific work of designs optimized for maximum specific work.

Figures 13-33 through 13-35 are identical to Figures 13-30 through Figure 13-32 except that their heat exchanger model is lossy. They reflect calculations in which the heat exchanger's pressure ratio was held fixed at 80 %.

The Figures 13-30 through 13-35 plots show that the optimized compressor pressure ratio drops as component losses increase, but that it drops more for turbine losses than for compressor losses. The approximate negative one slopes of cycle efficiency and specific work contours show that these quantities are sensitive to the average of the compressor and turbine polytropic efficiencies but are not very sensitive to their difference.

Textbook references indicate that turbomachine efficiencies in the range of 70% to 80% are common, although both higher and lower values are also in use. Inspection of Figures 13-30 through 13-35 shows that with such low values of turbomachine efficiencies, the compressor pressure ratio to optimize specific work for this temperature range would be between 10 and 30, reduced from the optimal ratio in the loss-free component case which exceeded 130. The specific work and cycle efficiency would be between one quarter and one half of the values calculated for the ideal case of Table 13-1.

As discussed before, there are nearby design points with slightly higher pressure ratios which yield a few percent higher efficiency at the cost of a small reduction in specific work. However, these designs are not very different from the optimum specific work designs. Larger increases in pressure ratio can reduce both efficiency and specific work.

These plots underline the importance of avoiding low component efficiencies in the design of nuclear engines for Mars. For example, Figure 13-31 shows near its lower left corner a *zero efficiency* contour. Thus, with a realistic heat exchanger pressure drop, any compressor and expander turbomachine component efficiencies below about 51% would result in developing no output power whatsoever from the engine, regardless of how much thermal power its reactor develops. **The plots also indicate that a large benefit would accrue from boosting turbomachine efficiencies upwards towards 90% instead of letting low pressure characteristics of the martian air drag them down towards 50% or even lower.**

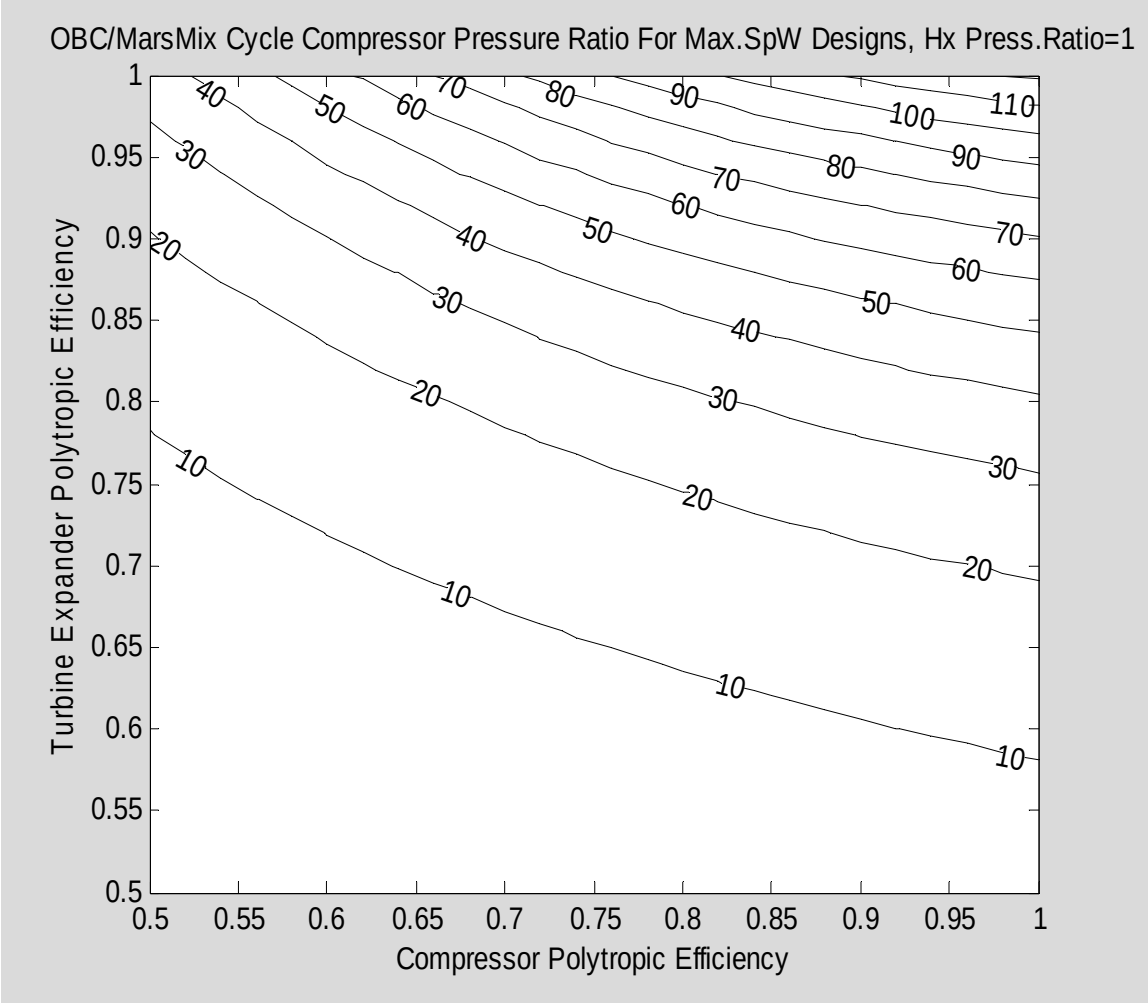


Figure 13-30: Optimized Compressor Pressure Ratio At Maximum Specific Work Design Points Vs Turbomachine Polytropic Efficiencies, For A Heat Exchanger Pressure Ratio Of 1.00 (T0=215, T4=1290 Kelvins)

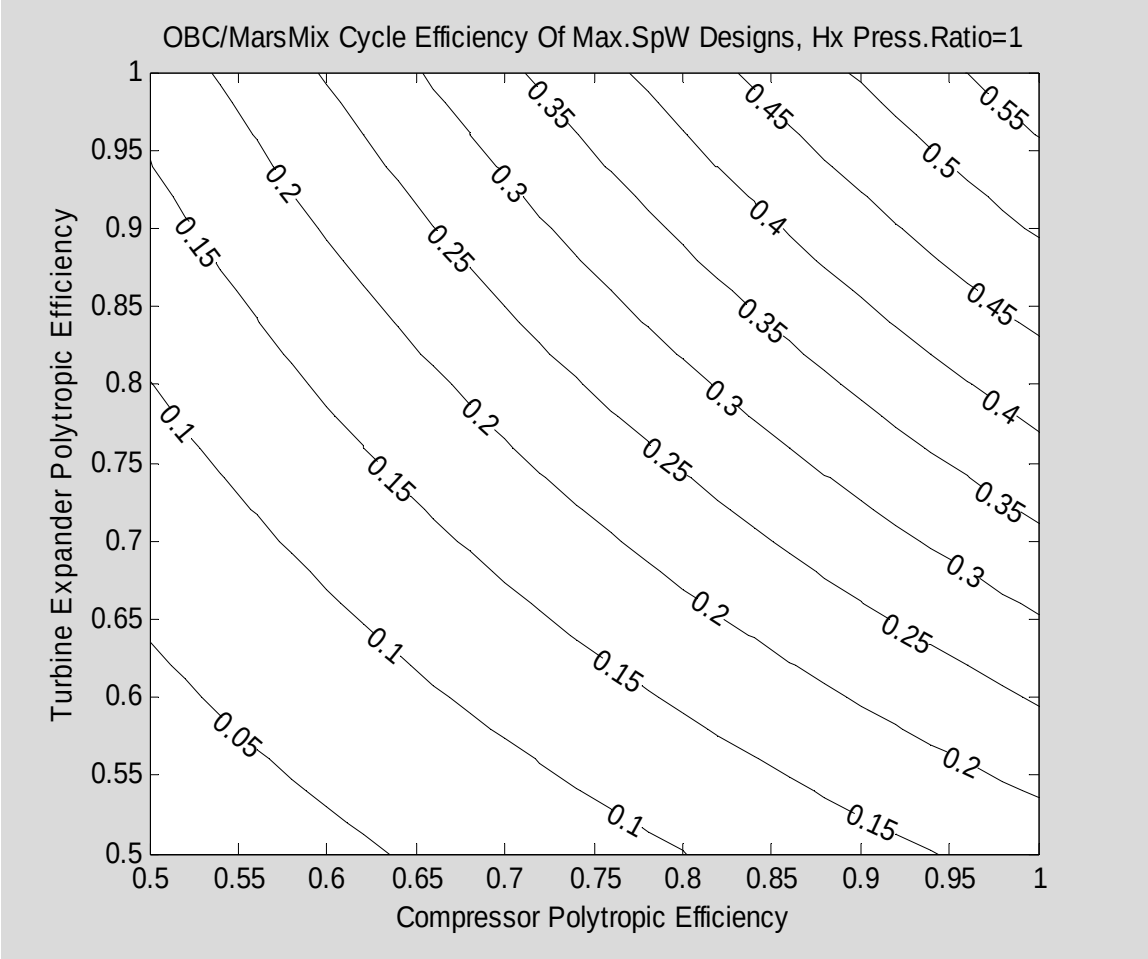


Figure 13-31: Cycle Efficiency At Maximum Specific Work Design Points Vs Turbomachine Polytropic Efficiencies, For A Heat Exchanger Pressure Ratio Of 1.00 (T0=215, T4=1290 Kelvins)

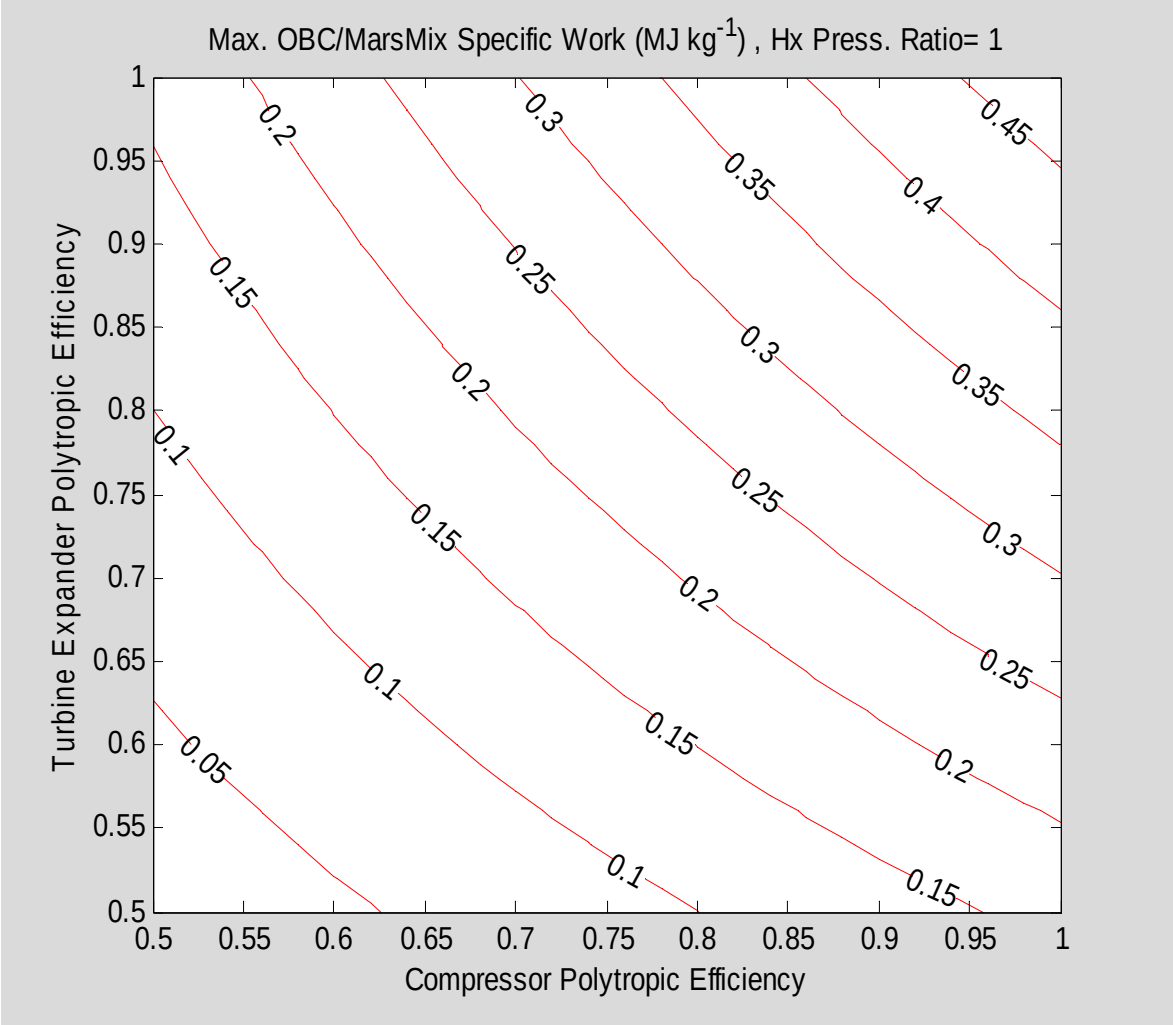


Figure 13-32: Specific Work At Maximum Specific Work Design Points Vs Turbomachine Polytropic Efficiencies, For A Heat Exchanger Pressure Ratio Of 1.00 (T0=215, T4=1290 Kelvins)

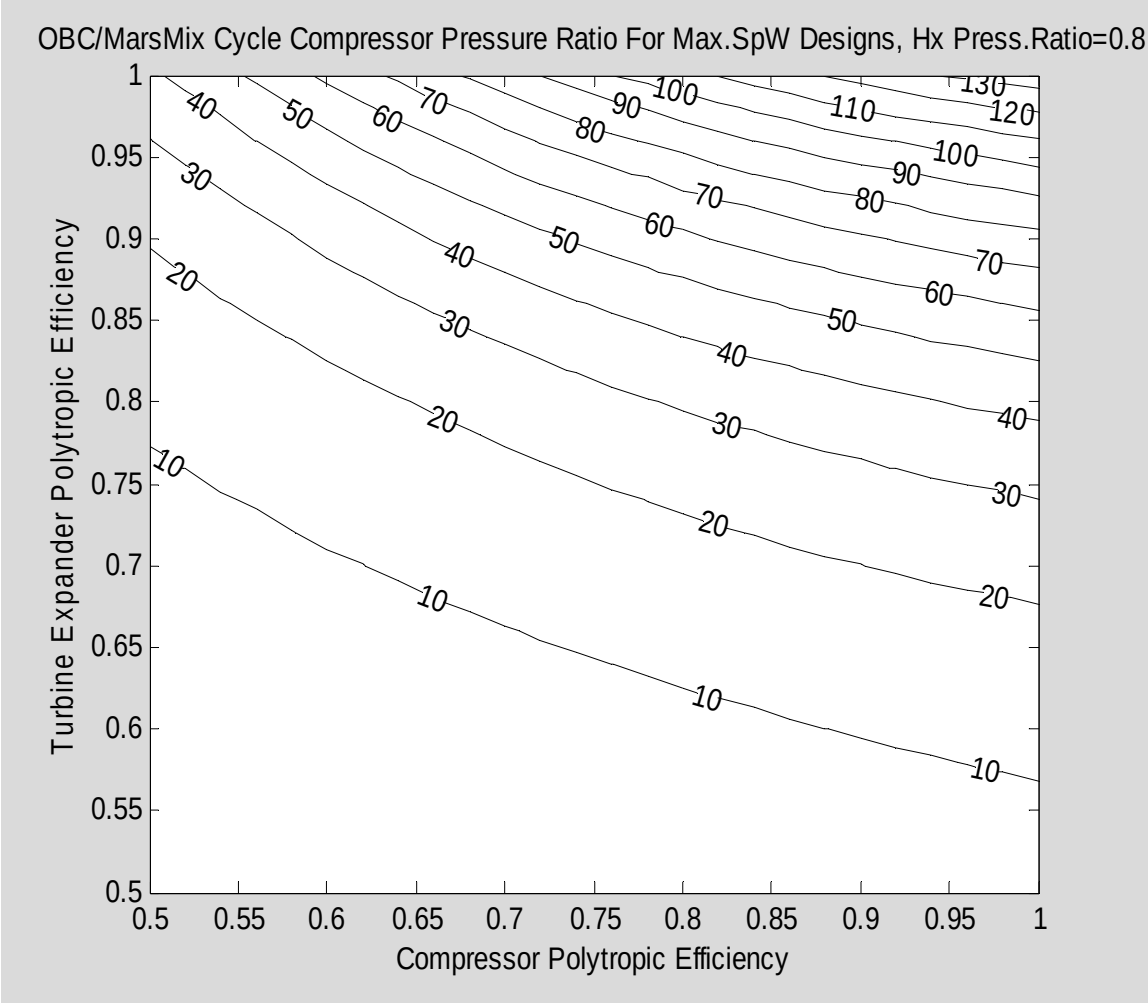


Figure 13-33: Optimized Compressor Pressure Ratio At Maximum Specific Work Design Points Vs Turbomachine Polytropic Efficiencies, For A Heat Exchanger Pressure Ratio Of 0.80 (T0=215, T4=1290 Kelvins)

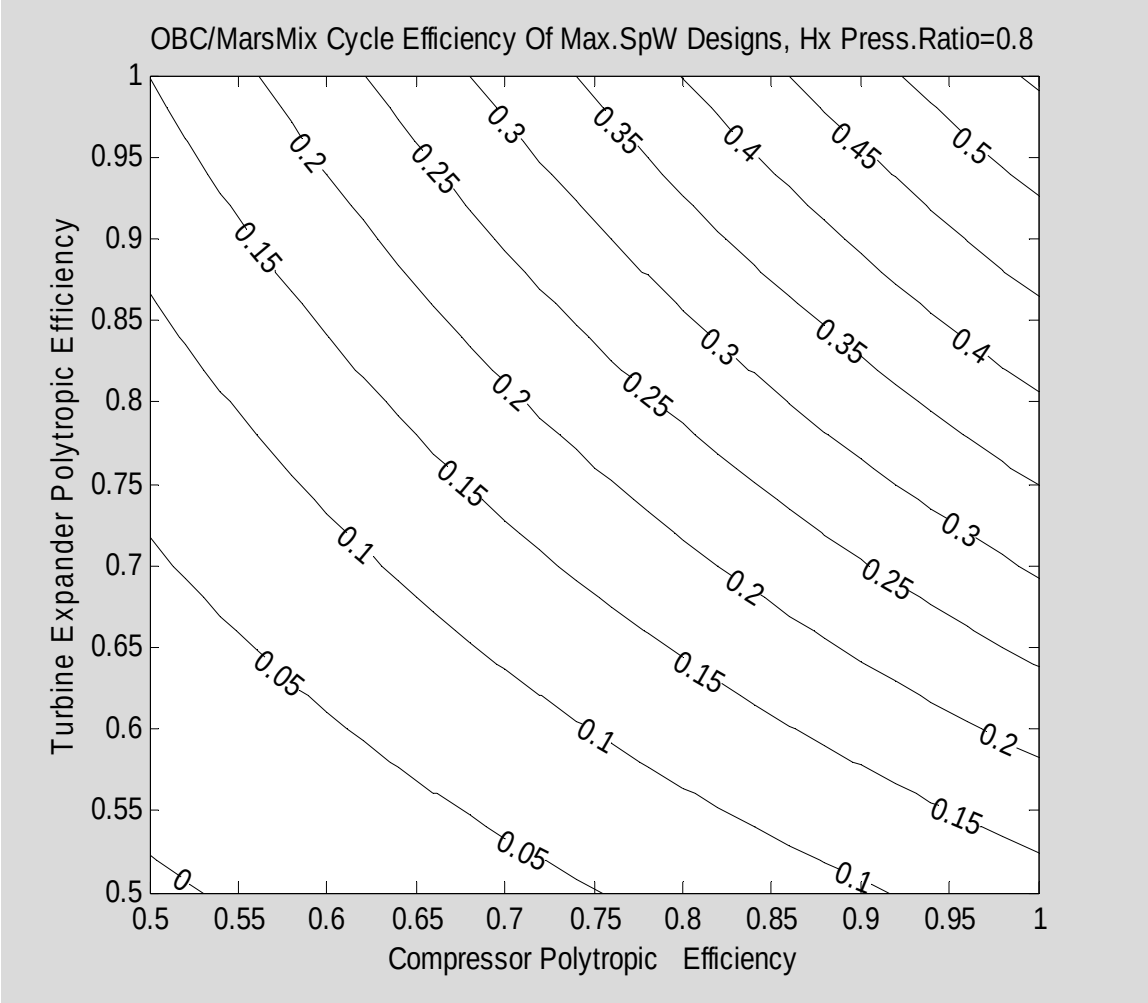


Figure 13-34: Cycle Efficiency At Maximum Specific Work Design Points Vs Turbomachine Polytropic Efficiencies, For A Heat Exchanger Pressure Ratio Of 0.80 (T0=215, T4=1290 Kelvins)

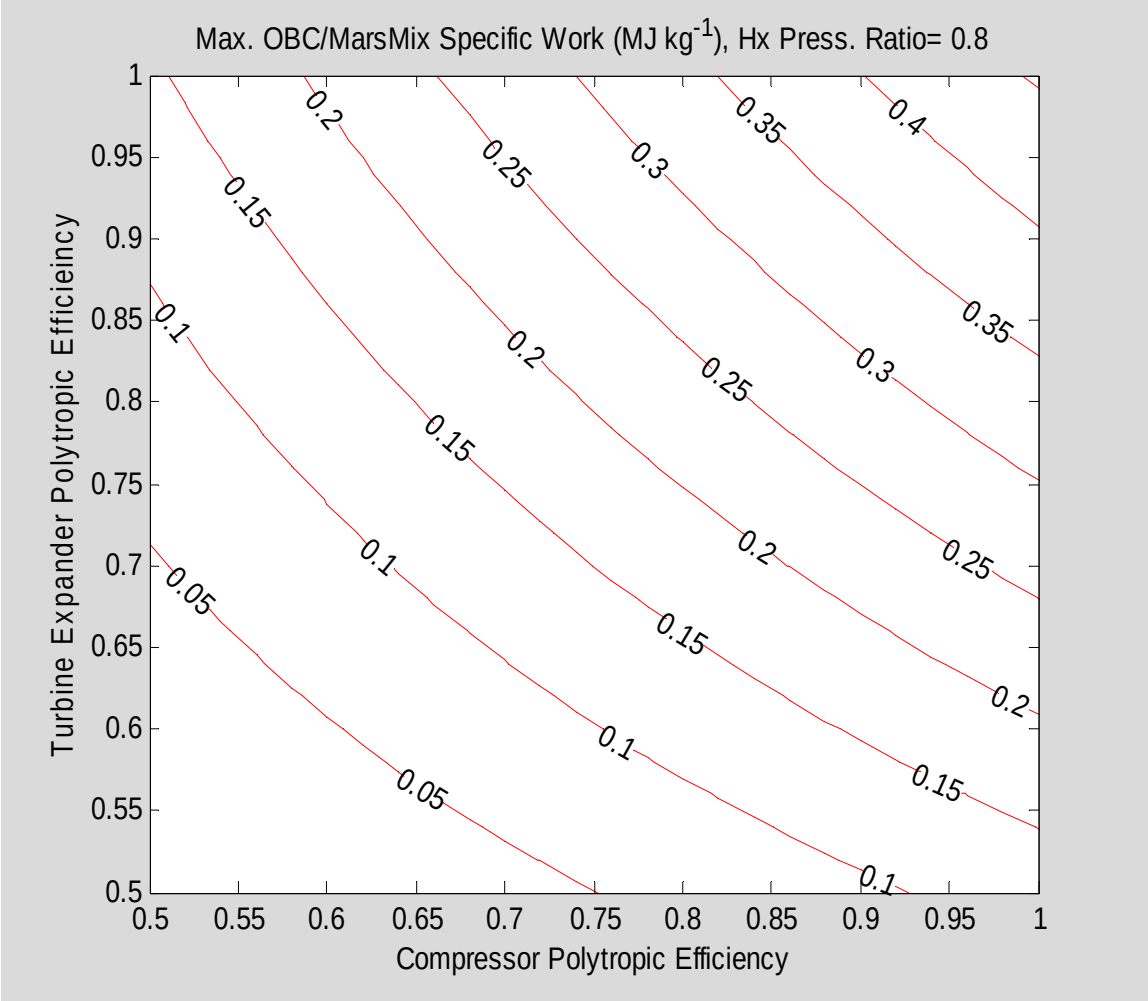


Figure 13-35: Specific Work At Maximum Specific Work Design Points Vs Turbomachine Polytropic Efficiencies, For A Heat Exchanger Pressure Ratio Of 1.00 (T0=215, T4=1290 Kelvins)

CHAPTER 14
14 TURBINE/COMPRESSOR THEORY AND PRACTICE

14.1 Introduction and Summary

Thermal energy conversion requires a compressible thermodynamic working fluid and the ability to convert between external work and internal fluid energy by fluid compression or expansion. Technological options for compressing or expanding fluids include (1) reciprocating displacement machines such as pistons and (2) fluid dynamic machines such as turbines. In displacement machines, timed valves periodically connect and disconnect a *trapped* internal volume to external manifolds holding incoming and outgoing working fluid, while the boundary surface enclosing the internal trapped fluid is oscillated to increase or decrease its volume in synchrony with the external valve operations. In fluid dynamic machines, the high speed moving surfaces of a continuously rotating impeller assembly produce fluid forces on continuously flowing *untrapped* fluid which cause the desired changes in fluid pressure. Within certain parameter ranges fluid dynamic machines can obtain higher energy efficiencies than displacement machines. The present day widespread use of fluid dynamic machines for energy conversion results from their high efficiencies and rugged simplicity and also from their lack of the reciprocating forces which increase fatigue failures of materials.

Today's use of fluid dynamic machines for thermal energy conversion is deeply rooted in the historical development of turbines and pumps for use with incompressible fluids, primarily water. It turns out that fluid dynamic machine designs developed for incompressible fluids can also be used with few modifications for compressible fluids. Although incompressible fluids do not allow thermal energy conversion, the extraction of hydro power from flowing river water has been used for the past 2000 years, e.g., to operate flour grinding mills. Efficiently extracting power from flowing water became an active focus of applied research during the 1700s. During the 1800s many fluid dynamic machines now in use were developed, and the work of Stokes, Navier, and Reynolds discovered the mathematical physics of fluid flow. Subsequent boundary layer work by Prandtl and others led in the early 1900s to approximate solutions of the fluid dynamic equations and to practical methods for calculating fluid dynamic forces. At the same time, application of the Buckingham Pi Theorem led to the widespread use of dimensionless parameters to characterize fluid dynamic applications. Dimensionless parameters allowed economical wind tunnel studies of small physical models to optimize larger applications, notably for the designs of aircraft wings, ship hulls, and various fluid dynamic machines such as aircraft propellers and turbines for aircraft jet engines. Because of the resulting technological advances, steam turbines replaced steam pistons in most fixed-site electric power plants during the early 1900s, and fluid dynamic machines later partially supplanted reciprocating displacement pumps as the Brayton gas cycle has been increasingly chosen over the Rankine steam cycle. Today, fluid dynamic machines for thermal conversion are an essential part of the global economy, with millions of

centrifugal compressors and turbines manufactured annually for automobile turbochargers alone.

Although the basic designs of fluid dynamic machines on today's market are quite old, even minor changes to the designs are prized if they confer a competitive advantage. Thus, the latest technological developments in fluid dynamic machines are proprietary and were not available for this present design effort. During the 1980s, numerical solution of Computational Fluid Dynamics (CFD) equations became a practical design alternative to physical model studies, thus graduating from the university research environments that had incubated these methods. CFD methods have since been embraced by industry as computer costs have dropped and computer capabilities have expanded. CFD methods would be useful in optimizing fluid dynamic machines for thermal conversion within a nuclear engine for Mars, but the best existing CFD codes for turbine and compressor design are likely the commercial properties of companies deriving their income from advanced turbine designs.

Because I did not have access to appropriate CFD codes and I lack experience and formal training in turbine design, the machine designs developed herein instead rely on past industry experience and on dimensional similitude as described in textbooks on turbomachinery design. The following textbooks have been heavily consulted:

(1) D.G.Shepherd, Principles Of Turbomachinery, The Macmillan Company, 1956, 463 pages

(2) G.T.Csanady, Theory of Turbomachines, McGraw-Hill Book Company, 1964, 376 pages

(3) O.E.Balje, Turbomachines, A Guide To Design, Selection, and Theory, John Wiley and Sons, 1981, 513 pages

(4) E.Logan, Jr, Turbomachinery, Basic Theory and Applications, 2nd Ed., Marcel Dekker,Inc., 1993, 261 pages

(5) R. Decher, Energy Conversion, Oxford University Press, 1994, 676 pages

The present chapter applies the methods of these textbooks and their included parametric design information (which is typically presented in graphical form) in order to select turbomachinery design parameters appropriate for a mobile martian nuclear engine's Open Brayton Cycle thermal conversion system. In essence, this is equivalent to choosing between existing designs, then adjusting the choice for the martian application.

14.2 Types of Turbomachines

Many standard turbomachine designs have been developed historically and are still in common use. Each standard design employs a rotor, sometimes called an impeller, which rotates as a rigid body about an axle. Designs differ in the shapes of their rotors, in the shapes of stationary fluid-guiding surfaces which they employ outside the rotors, and in the designed schemes for the working fluid to flow between rotating and stationary structures.

The many types of turbomachines are broadly classified into two major categories, radial flow machines and axial flow machines, according to whether the fluid flow through the rotor is primarily in the radial or the axial direction. Some types such as Francis and mixed-flow machines combine both, but most other types are clearly axial or radial designs. Figure 14-1 from Decher 1994 [p 481] who obtained the diagram from an earlier reference) shows axial flow compressors typical in aircraft propulsion or in ground-based Brayton-cycle gas power plants. In the upper diagram all the rotating airfoil blades on the rotor are rooted to the same shaft and therefore rotate together. Rows of nonrotating stationary airfoil blades are mounted on the housing interleaved between rotating blade rows, so that flowing air alternately encounters rotating and nonrotating airfoil blades. Successive rotating blade rows have varying blade lengths and shapes which are optimized for the amount of gas compression accumulated prior to encountering each row. The pressure ratio due to compression in each blade row is typically between 1.1 and 1.2 [Decher 1994, 479] The lower diagram shows a 2-spool compressor in which the use of two concentric shafts rotating at different angular speeds allows less variation in aerofoil blade length for the same overall compressor pressure ratio. Axial flow turbomachines were typically of lower efficiency than radial flow machines in the mid 1900s but surpassed radial machines as a result of the enormous research investment devoted to aircraft propulsion. However, there is no intrinsic reason for axial machines to be superior to radial machines.

Figure 14-2 (also from Decher) shows a radial flow air compressor with dual suction air intakes. Air enters radially through both the front and rear air intake casings which are labeled in the diagram, and on each side is guided by stationary swirl vanes followed by intake chutes to turn and enter the rotor axially through an eye near the rotor's center. The primary scheme is that radial vane blades within its rotor entrain the air so that it rotates with the rotor and is driven outwards by centrifugal force. High speed radially exiting air is slowed within the surrounding volute casing region (not shown in Figure 14-2), then exits at high pressure through air outlet casing pipes. The pressure ratio from a single-stage radial compressor is typically between 3 and 10 [Decher 1994, 479].

Because of the low pressure of the martian air it will be necessary to use multiple stages to reach pressures for which thermal energy conversion becomes practical. Radial flow machines are superior for this application because they provide higher pressure ratios per stage. They also have more air intake area per unit volume than axial turbines, which is useful in low density air. Furthermore their diameters are larger for a given volume which tends to reduce viscous losses due to Reynolds Number effects, as discussed later. For these reasons, radial flow turbomachines, i.e., centrifugal compressors and radial inflow turbines, are chosen for use in the present design.

14.3 Dimensional And Dimensionless Turbomachine Parameters

Table 14-1 lists the primary variables traditionally used to characterize turbomachine applications. Some explanatory comments follow.

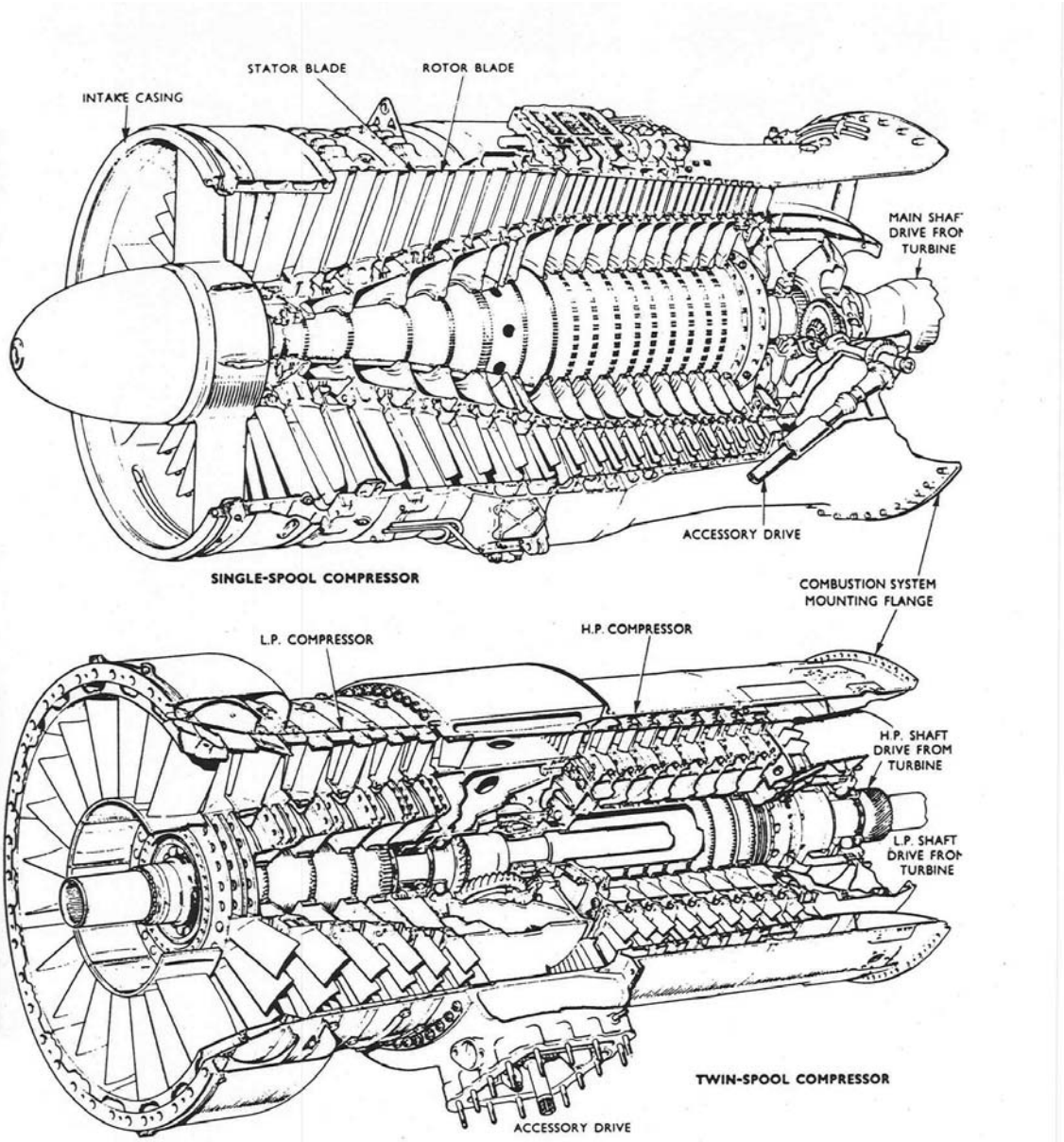


Figure 14-1: Axial Flow Compressors (from [Decher])

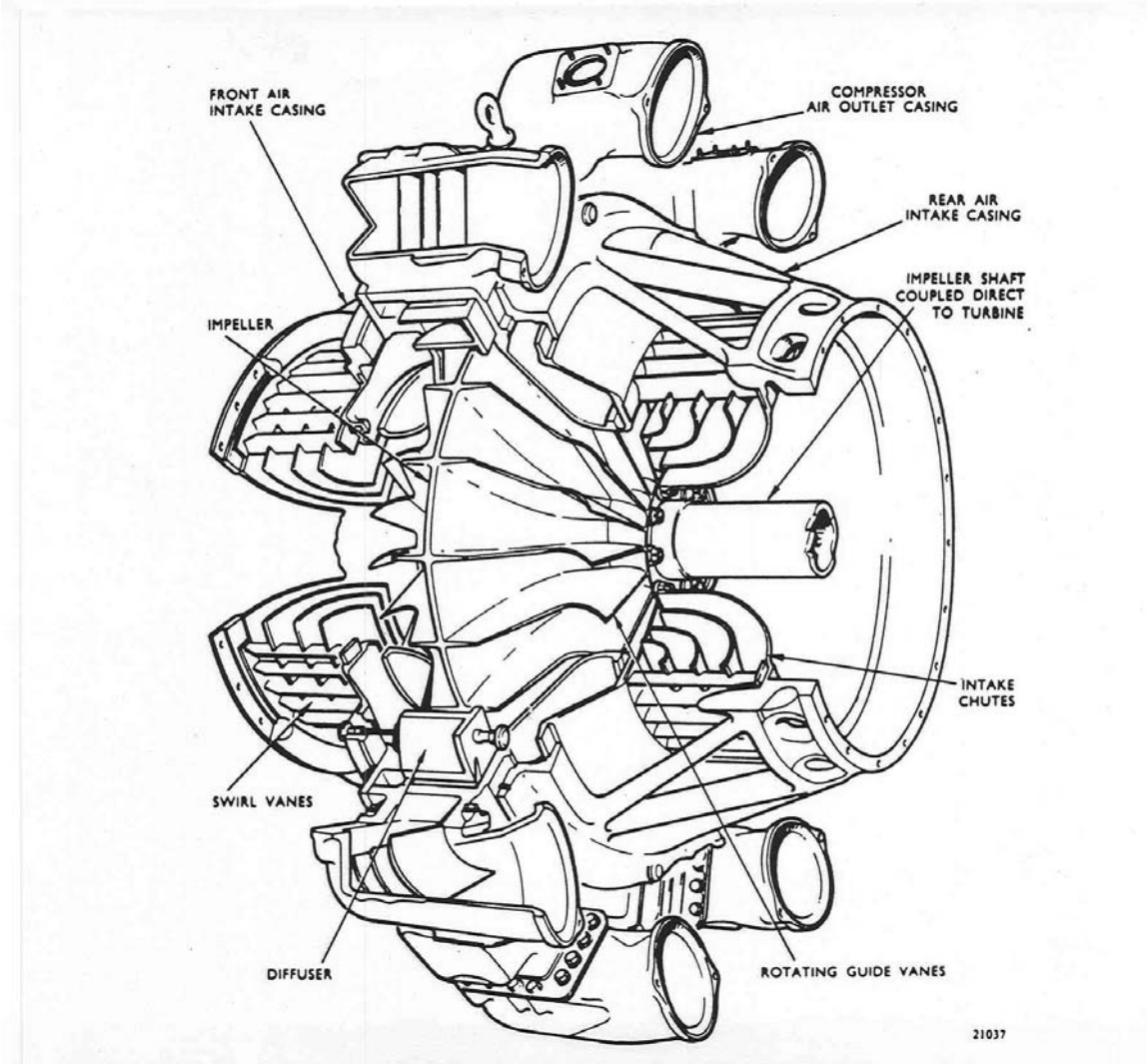


Figure 14-2: Dual Radial Flow Compressors (from [Decher])

Table 14-1: Dimensional Parameters Characterizing Turbomachine Applications

Parameter	Typical Symbols	Typical Metric Units	Unit Dimensions
(Angular) Speed	N	radians/second	$(Time)^{-1}$
Rotor Diameter	D	meters	(Length)
Volumetric Flowrate	Q, or V	$meter^3$/second	$(Length)^3(Time)^{-1}$
Energy Transfer, or Head	E, or gH	joules/kg	$(Length)^2(Time)^{-2}$
Kinematic Viscosity	ν $(=\mu/\rho)$	$meter^2$/second	$(Length)^2(Time)^{-1}$
Fluid Mass Density	ρ, or γ	$kg/meter^3$	$(Mass)(Length)^{-3}$
Fluid Viscosity	μ	pascal-seconds	$(Mass)(Length)^{-1}(Time)^{-1}$
Ratio of Specific Heats	κ, or γ	(none)	(none)
Fluid Pressure	p	pascals	$(Mass)(Length)^{-1}(Time)^{-2}$
Sound Speed	c_s	meters/second	$(Length)(Time)^{-1}$
Critical Speed	c_s*	meters/second	$(Length)(Time)^{-1}$
Power	P, or W	watts	$(Mass)(Length)^2(Time)^{-3}$

The speed or angular speed, N, is sometimes quoted in revolutions per second or revolutions per minute. However, it must be expressed in *radians* per unit time in order to be consistent with existing *dimensionless* correlations of machine performance.

Although the volumetric flowrate, Q, is well-defined for applications in which the working fluid is incompressible, the volumetric flowrates at the inlet and the outlet of a machine are different for compressible fluids. By convention, the larger of the two volumetric flowrates is used in such cases. Thus, by convention Q represents the outlet volumetric flowrate for a turbine and the inlet volumetric flow rate for a compressor.

The traditional head parameter for fluid dynamic machines, H, as listed in Table 14-1, is the reversible work energy transferred per unit *weight* of fluid instead of per unit mass, so it is traditionally measured as a height, i.e., in distance units. Multiplication of H by g, the acceleration of gravity at Earth's surface, converts from head to E, the reversible work energy transferred in the turbomachine per unit fluid mass. If the working fluid is also incompressible, then the head, H, multiplied by the fluid's *weight density*, or equivalently the reversible work energy, E, multiplied by the fluid's *mass density*, equals the pressure difference between the machine's inlet and outlet fluid. The situation is more complex for compressible fluids for which part of the work energy changes the fluid's density and temperature. For compressible fluids, E equals the change in fluid enthalpy that would have resulted if the turbomachine's pressure change had instead occurred isentropically. It can be calculated for compressors and turbines by using, respectively, Eqs 13-40b and 13-42b to evaluate enthalpies of the hypothetical states that would have been reached isentropically.

It is not uncommon to encounter machine parameter data quoted in non-metric units such as cubic feet per minute, inches of head, pounds per square inch of pressure, or viscosities in poise. Any such data have been converted to metric units if used herein.

The first four dimensional parameters in Table 14-1, N, D, Q, and E are always appropriate for any turbomachine application, regardless of whether its working fluid is an incompressible liquid or a compressible gas. The Buckingham pi theorem asserts that from these four dimensional parameters only two *independent* dimensionless groupings can be formed, but that pair of dimensionless values can be defined in various ways. The most common traditional dimensionless combination is the *flow number*, ϕ, sometimes called the *flow coefficient*, defined as follows:

$$\phi = \frac{Q}{ND^3} \tag{14-1}$$

Another common traditional dimensionless combination, independent from ϕ, is the *head number*, ψ, sometimes called the *head coefficient*, which is defined as follows:

$$\psi = \frac{E}{N^2 D^2} \tag{14-2}$$

Other useful and traditional dimensionless combinations of the first four dimensional parameters in Table 14-1 can either be expressed in terms of ϕ and ψ or determined directly from N, D, Q, and E. The most important of these dimensionless groupings is the *specific speed,* N_s , which is defined as follows:

$$N_s = \frac{\phi^{1/2}}{\psi^{3/4}} = \frac{NQ^{0.50}}{E^{0.75}} = \Omega \tag{14-3}$$

Some authors (e.g., [Csanady 1964, 15]) denote the specific speed with the symbol, Ω. The extensively used dimensionless *specific diameter*, D_s, is defined as follows:

$$D_s = \frac{\psi^{1/4}}{\phi^{1/2}} = \frac{DE^{0.25}}{Q^{0.50}} = \Delta \tag{14-4}$$

Some authors (e.g., [Csanady 1964, 20]) denote the specific diameter with the symbol, Δ. Another sometimes encountered dimensionless grouping of the same dimensional parameters, N, D, Q, and E, is the *power coefficient*,

$$C_P = \phi\psi = \frac{QE}{N^3D^5} \tag{14-5}$$

However, these five dimensionless parameters are clearly not independent, since the last three can be expressed as functions of the first two. Actually, any two can be selected as fundamental and the others expressed in terms of them. The most common classification of turbomachines, the Cordier Diagram discussed later, selects the specific speed, Ns, and the specific diameter, Ds, as its pair of basic dimensionless parameters.

Fluid viscosity causes friction so one would expect that including viscous effects could lead to machine correlations predicting efficiency. Kinematic viscosity is defined as the ratio of the fluid's viscosity to the fluid's mass density. By adding the kinematic viscosity to the first four dimensional variables a third independent dimensionless parameter may be formed. Although this third independent grouping could be chosen in various ways, generally accepted convention calls it the Machine Reynolds Number, Re*. Herein it is defined as follows:

$$\mathrm{Re}^* \equiv \frac{ND^2}{2\nu} \equiv \frac{v_{tip}D}{\nu} \equiv \frac{\rho ND^2}{2\mu} \tag{14-6}$$

Note that some authors (Csanady, Logan) omit the factor of 2 in the denominator, thus defining the machine Reynolds number as ND^2/ν. The factor of 2 is included herein to be consistent with definitions by Shepard and by Balje which use the rotor tip speed. Because of this discrepancy some care must be taken in interpreting graphical performance data plotted versus machine Reynolds number as included in the texts.

Thus, high kinematic viscosity of a working fluid is traditionally represented as a low value of the Machine Reynolds Number, Re*, which can also be produced by employing small values of D or N. Although many applications are relatively independent of Re*

effects, the Re* effects are important in applications involving very small turbomachines, or high viscosity fluids such as viscous oils **or low density gases like martian ai**r.

Gas viscosity is almost completely independent of pressure for the pressure range expected within the nuclear engine. However, viscosity does vary with temperature. Figure 14-3 plots the viscosity of MarsMix gas versus temperature, as calculated from tabulated viscosities of each of its constituents.

On the other hand, the kinematic viscosity, which is the ratio of viscosity to mass density, depends on both temperature and pressure via Eq. 14-7. Figure 14-4 indicates how the kinematic viscosity of martian air varies within the envelope of conditions that might be encountered within designed engines.

The parameters discussed so far could apply equally well to either compressible or incompressible fluids, but thermal conversion systems require the working fluid to be compressible. A compressible fluid has other important behaviors, notably its conversion between mechanical and thermal energy and its sound wave propagation. An adequate model of the present design's martian air working fluid is as an ideal and perfect gas with non-constant specific heats, which some authors choose to call a semiperfect gas. For any ideal gas, the fluid mass density is given by an ideal gas law function of its pressure and temperature, i.e.,

$$\rho = \frac{p}{RT} \tag{14-7}$$

where ρ represents gas density (kg/m^3), p represents gas pressure (Pa), T represents the absolute temperature of the gas (Kelvins) and R is the gas constant given by

$$R = \frac{\mathcal{R}}{\hat{M}} \tag{14-8}$$

where $\mathcal{R}$ is the universal gas constant, 8.31361 J/mole/Kelvin, and $\hat{M}$ is the molal mass of the gas. For martian air the molal mass is

$$\hat{M}_{\text{MartianAir}} = 0.04334 \text{ kg/mole}$$

so

$$R_{\textbf{MartianAir}} = 191.06 \text{ J kg}^{-1} \text{ Kelvin}^{-1} \tag{14-9}$$

For an ideal gas the internal energy, u, the enthalpy, h, the specific heat at constant pressure, c_p, and the specific heat at constant volume, c_v , are each functions of gas temperature alone, i.e., they are each independent of gas pressure. The ratio of the two specific heats is a crucially important dimensionless grouping for turbomachines since it governs gas heating and cooling by adiabatic compression and expansion:

$$\kappa = \frac{c_p}{c_v} \tag{14-10}$$

For many applications discussed in texts, κ is assumed to be a constant value. However, for the semiperfect martian air the two specific heats and thus κ, their ratio, vary

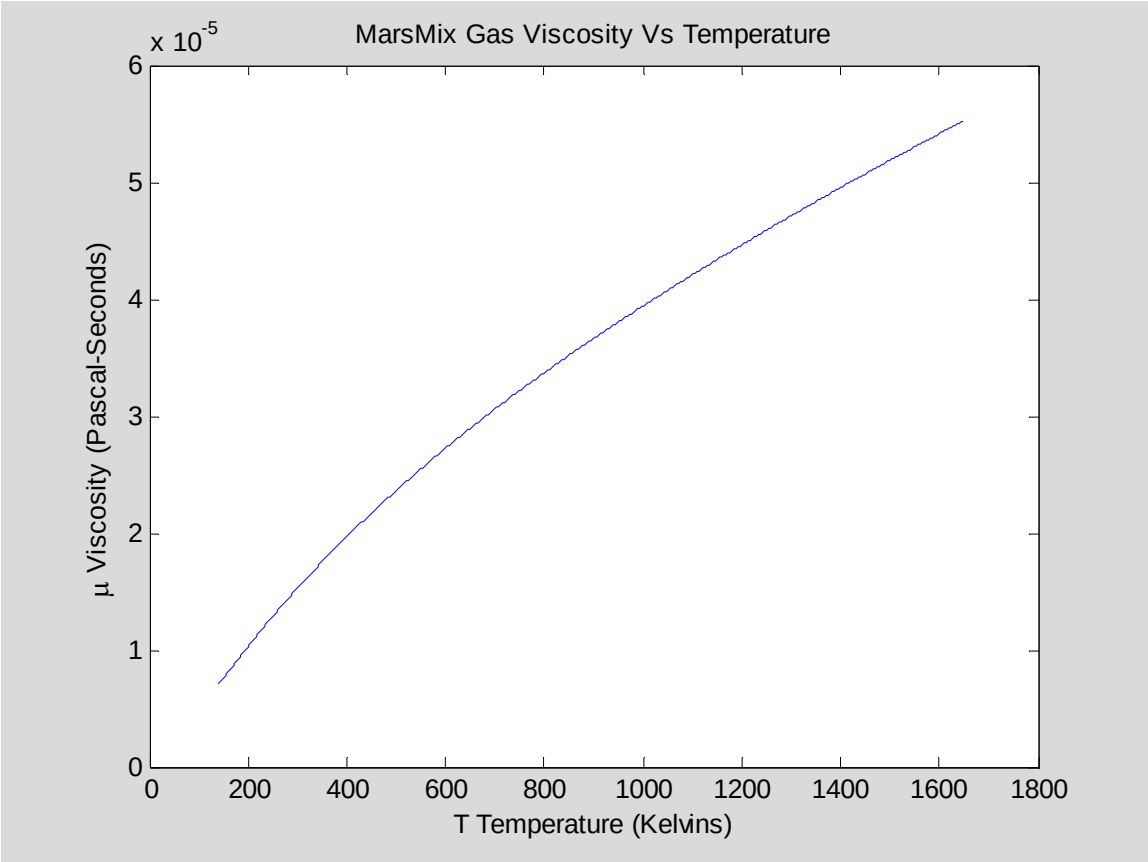


Figure 14-3: Calculated Viscosity Of MarsMix Gas (Simulant Of Martian Air)

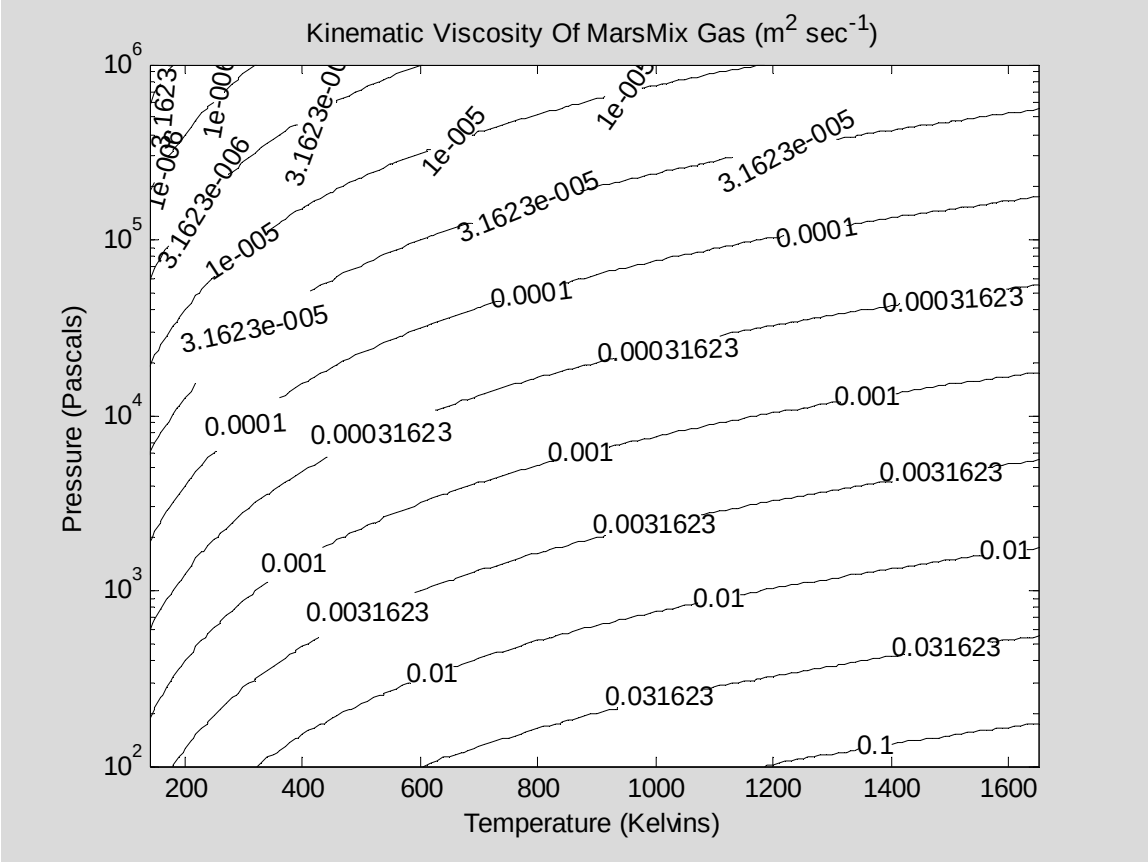


Figure 14-4: Calculated Kinematic Viscosity Of MarsMix Gas (Simulant Of Martian Air)

substantially over the full range of martian air temperatures expected within the engine. On the other hand, since the turbomachines within the engine will need to be configured in successive stages each of which encounters only a fraction of the total temperature range, for some analyses it may be sufficiently accurate to approximate κ by fixed constant values over the temperature *subranges* encountered within a turbomachine stage.

For an ideal gas the speed of sound is completely determined by the specific heat ratio, κ, the gas constant, R, and the absolute gas temperature, T, as follows:

$$c_s = \sqrt{\kappa R T} \tag{14-11}$$

For MarsMix gas, the speed of sound varies with temperature by about a factor of three over our full temperature range of interest, i.e., the range of temperatures which could be encountered at different locations within the engine. Eq. 14-11 applies but the specific heat ratio, κ, varies with temperature since the specific heats are not constant. The calculated sound speed vs temperature is plotted in Figure 14-5.

Many turbomachines operating with a gas rotate their impeller tips faster than the speed of sound, but most also limit rotor speeds *relative* to the flowing gas below sonic speeds to avoid internal shock waves. Texts indicate fully supersonic designs are less common since their efficient performance is more sensitive to design details. For machine applications involving high speed gas flows the Inlet Mach Number is a frequently encountered dimensionless parameter. It is the ratio of twice the rotor's tip speed to the speed of sound in the machine's inlet gas stream, calculated as follows:

$$M_{Inlet}* = \frac{ND}{c_s^{(Inlet)}} = \frac{ND}{\sqrt{RT_{Inlet}\kappa(T_{Inlet})}} \tag{14-12}$$

This is not the only Mach number in common use since other gas flows within a machine are also used to define different Mach numbers. Some authors (e.g., [Balje 1981,1]) prefer to not use a Mach number and instead use the Peripheral Machine Laval Number, La*, which is defined as the ratio of rotor tip speed to a critical velocity, c_s*, i.e.

$$La*_{(Inlet)} = \frac{ND/2}{c_s*(Inlet)} \tag{14-13}$$

This critical velocity is defined as follows:

$$c_s* \equiv \sqrt{\frac{2\kappa(T_t)}{\kappa(T_t)+1} RT_t} \tag{14-14}$$

where T_t is the *total* temperature at the machine inlet, i.e., the temperature which the moving air would reach if it were brought to rest adiabatically.

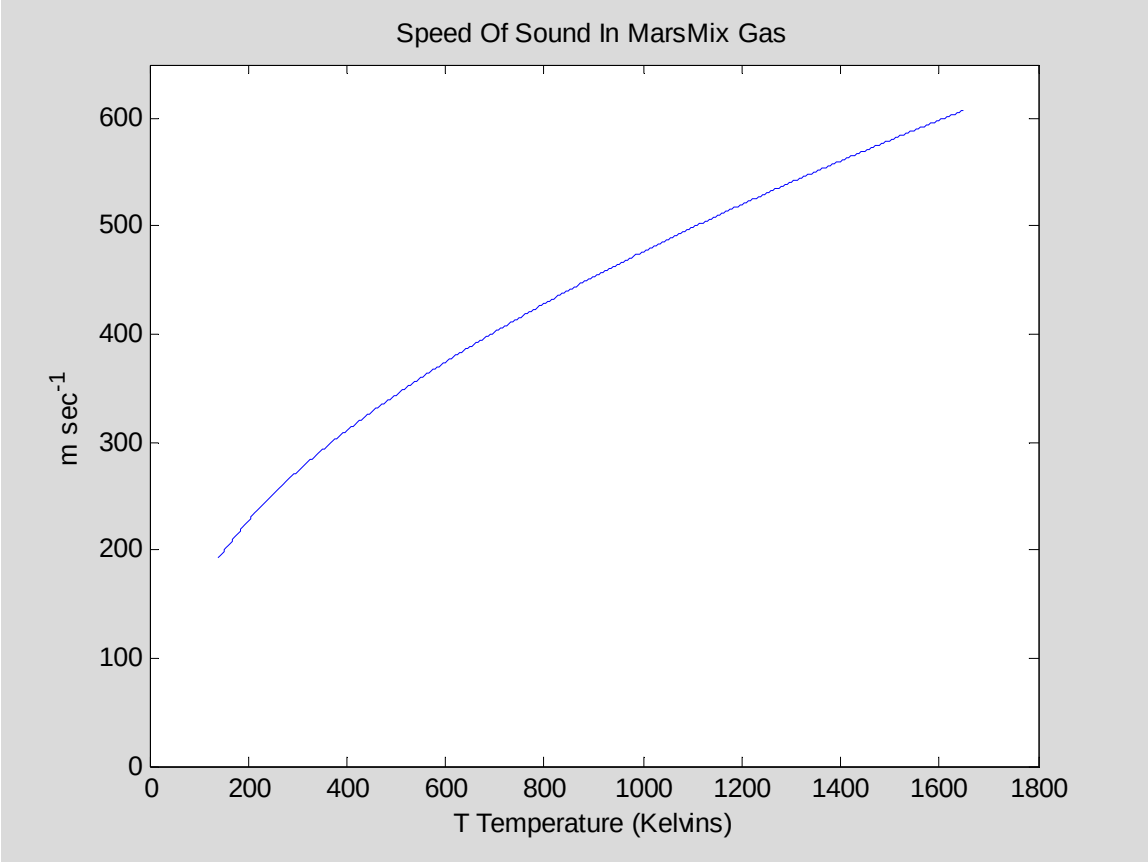


Figure 14-5: Calculated Sound Speed In MarsMix Gas (Simulant Of Martian Air)

14.4 Turbomachine Similitude And Associated Parametric Design Data

Although Shepherd's text discusses turbomachine similitude, the 1964 book by Csanady was the first textbook to publish a Cordier Diagram. It stated the following:

> "A particular popular presentation is due to Cordier who compiled an empirical $\Omega=f(\Delta)$ graph and showed that the scatter of points for high efficiency machines was relatively little. The Cordier diagram….may be used in deciding, say, the size of a machine required for a given duty."

Cordier had surveyed the parameters of many turbomachines then in operation, determining for each machine its energy efficiency, its Specific Speed, N_s, which he called Ω, and its Specific Diameter D_s, which he called Δ. He plotted the resulting (Δ,Ω) ordered pairs on log-log graph paper, observed that the highest turbomachine efficiencies were associated with (Δ, Ω) points forming a narrow band, then drew a smooth line following the middle of that high efficiency band. The useful significance of the Cordier Diagram is that one can be confident a highly efficient turbomachine design can be obtained for any points chosen close to the Cordier line. The referenced work by Cordier had been published in 1953 in *Brennstoff-Warme-Kraft*, so the machine designs it compiled may have been surpassed by 1964 and are even more likely obsolete by now. Thus, the Cordier diagram may possibly be too conservative, but it is a convenient tool. Csanady's published diagram from Cordier is reproduced as Figure 14-6.

As explained by Csanady, machine designs to the lower left of the high efficiency Cordier curve were possible but were not frequently deployed because they were usually not competitive in the marketplace except for certain special niche applications. Designs much to the upper right of the curve were simply not possible at the time.

Balje's 1981 text reports on his own similar but later survey of deployed turbomachine designs. In his own Cordier diagrams he prefers to swap axes from Csanady's graph, so that Balje's Specific Speed, n_s, appears on the horizontal axis while Specific Diameter, d_s, is on the vertical axis. Figure 14-7 reproduces two of Balje's Cordier Diagrams from his pages 58 and 59 showing some individual machine data points. In the upper diagram, Balje's annotations show the regions in which different machine type classifications usually fall, and he also indicates some associated efficiency numbers.

Instead of simply drawing a line through the higher efficiency designs, Balje's text further analyzes the survey data so that achieved efficiencies are graphically presented as contour plots. Figure 14-8 reproduces contour plots [Balje 1981, 48, 51] from his book showing the achieved turbomachine efficiencies for respectively compressors and turbines, each plotted versus (n_s,d_s) axes in what is now called a "Balje diagram".

To help in selection of turbomachine type for a design, Balje also presents (on page 55), the best efficiencies achieved for each machine type plotted as functions of n_s alone,i.e., with d_s suppressed. These plots are reproduced here as Figures 14-9 and 14-10.

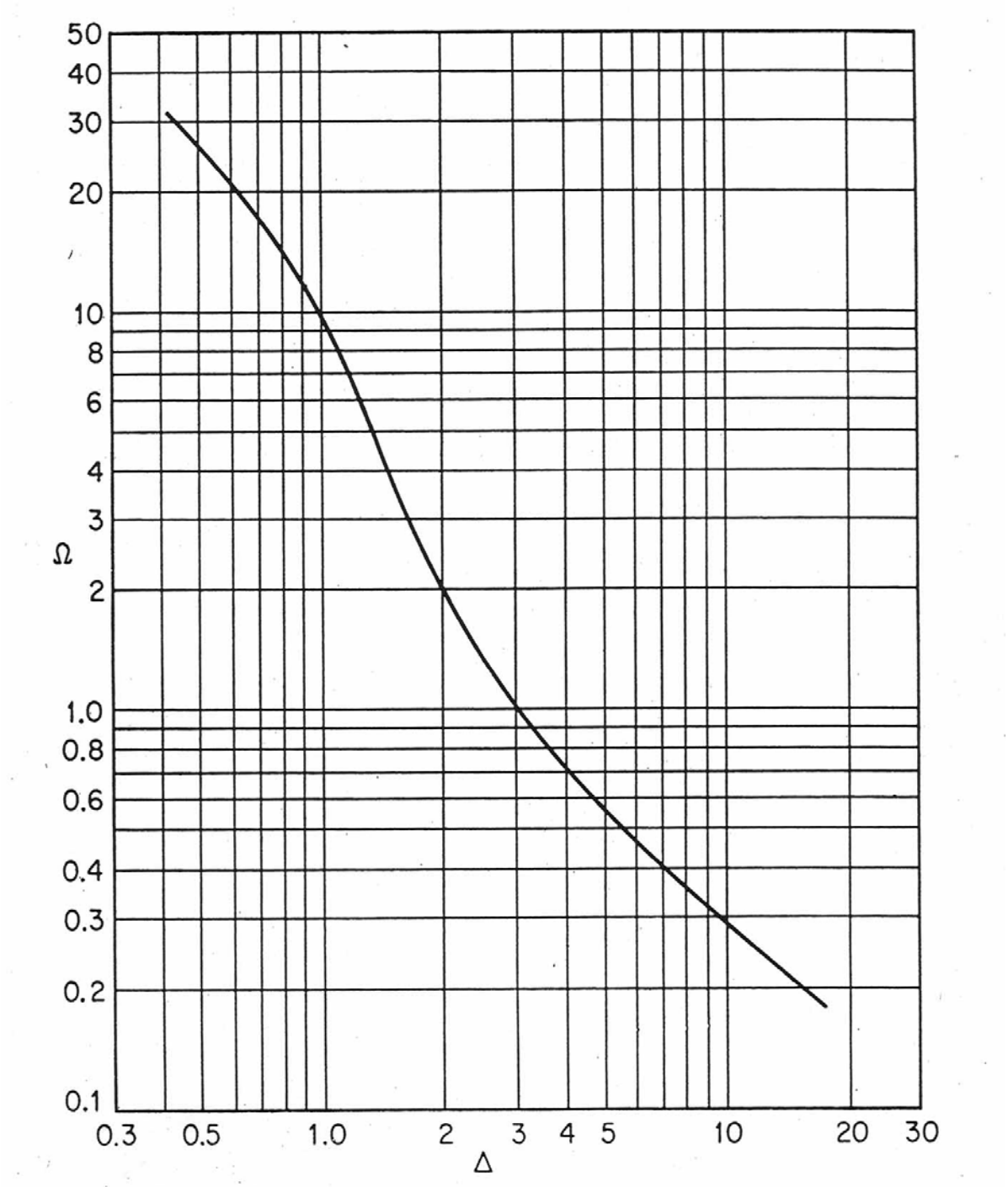


Figure 14-6: Cordier Diagram (From [Csanady])

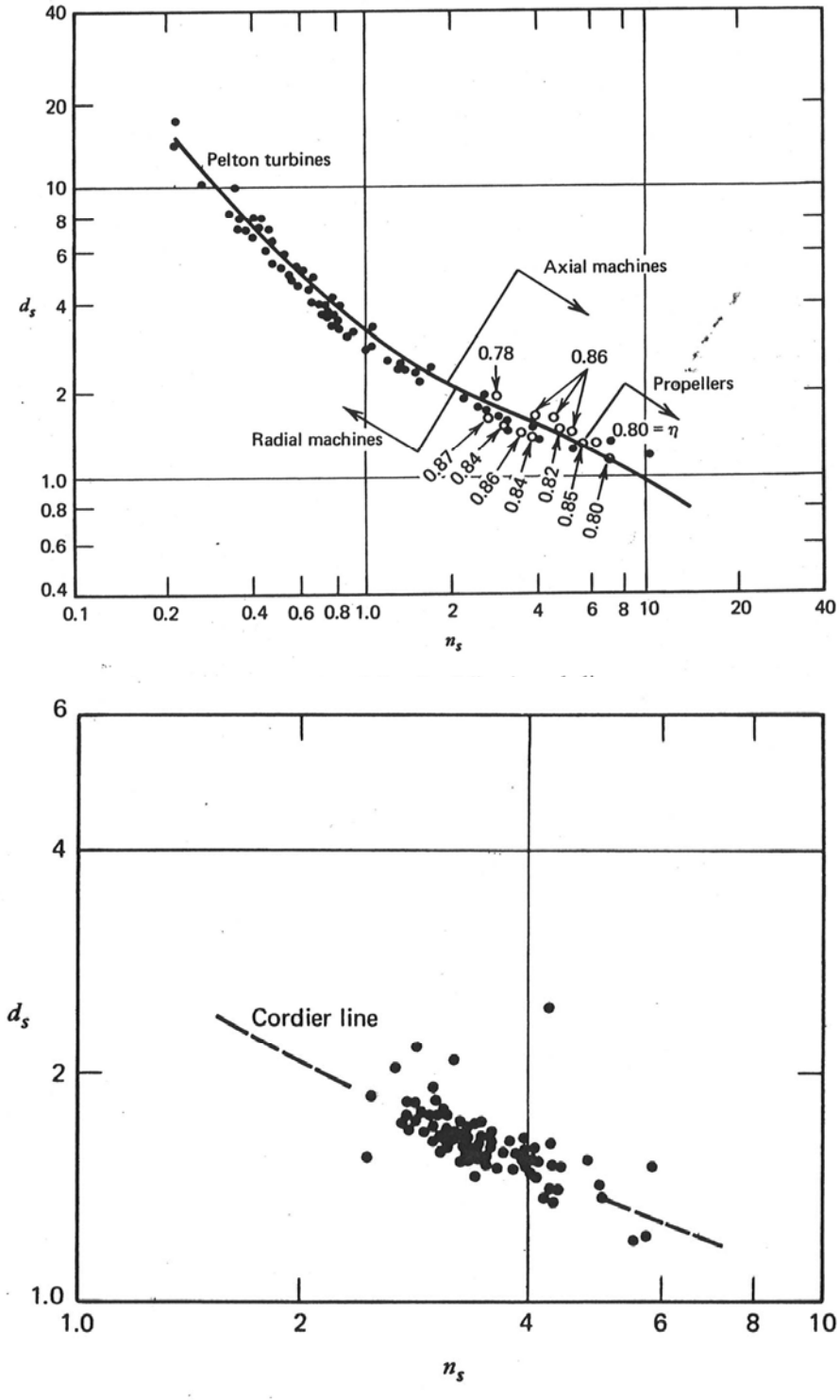


Figure 14-7 (a) and (b): Cordier Diagrams (from [Balje])

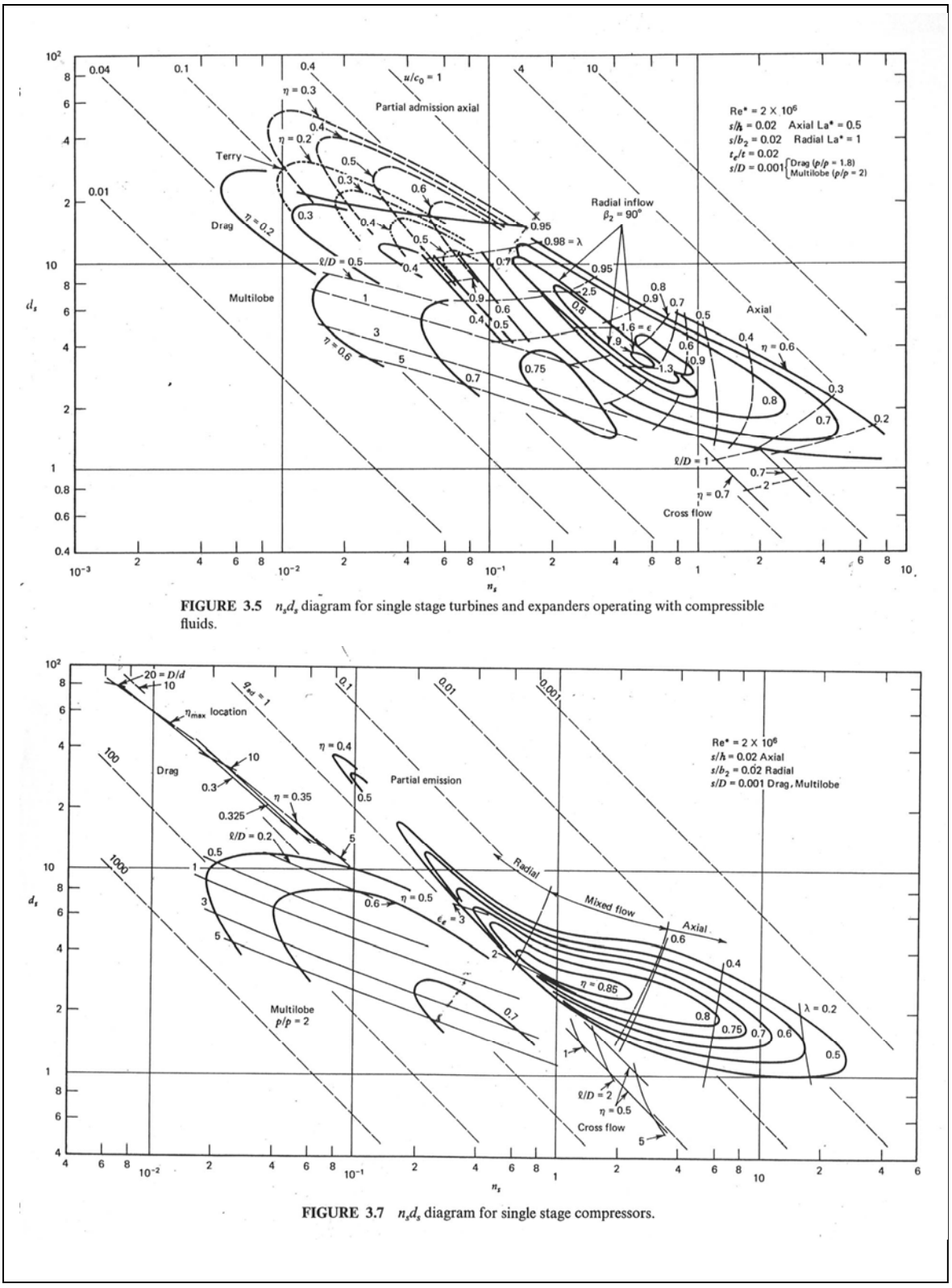


FIGURE 3.5 $n_s d_s$ diagram for single stage turbines and expanders operating with compressible fluids.

FIGURE 3.7 $n_s d_s$ diagram for single stage compressors.

Figure 14-8: (a) and (b): Achieved Turbomachine Efficiencies η vs (n_s, d_s) (from [Balje])

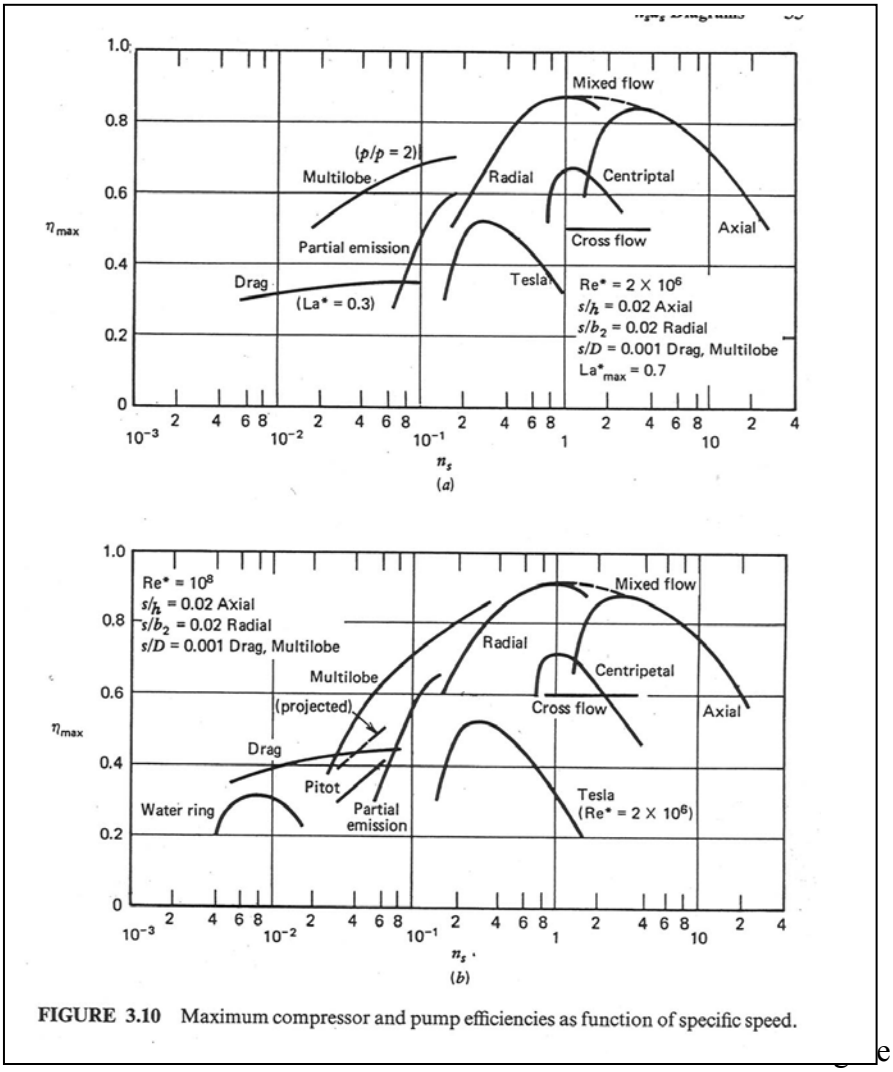


FIGURE 3.10 Maximum compressor and pump efficiencies as function of specific speed.

Figure 14-9: Maximum Compressor Efficiencies (from [Balje])

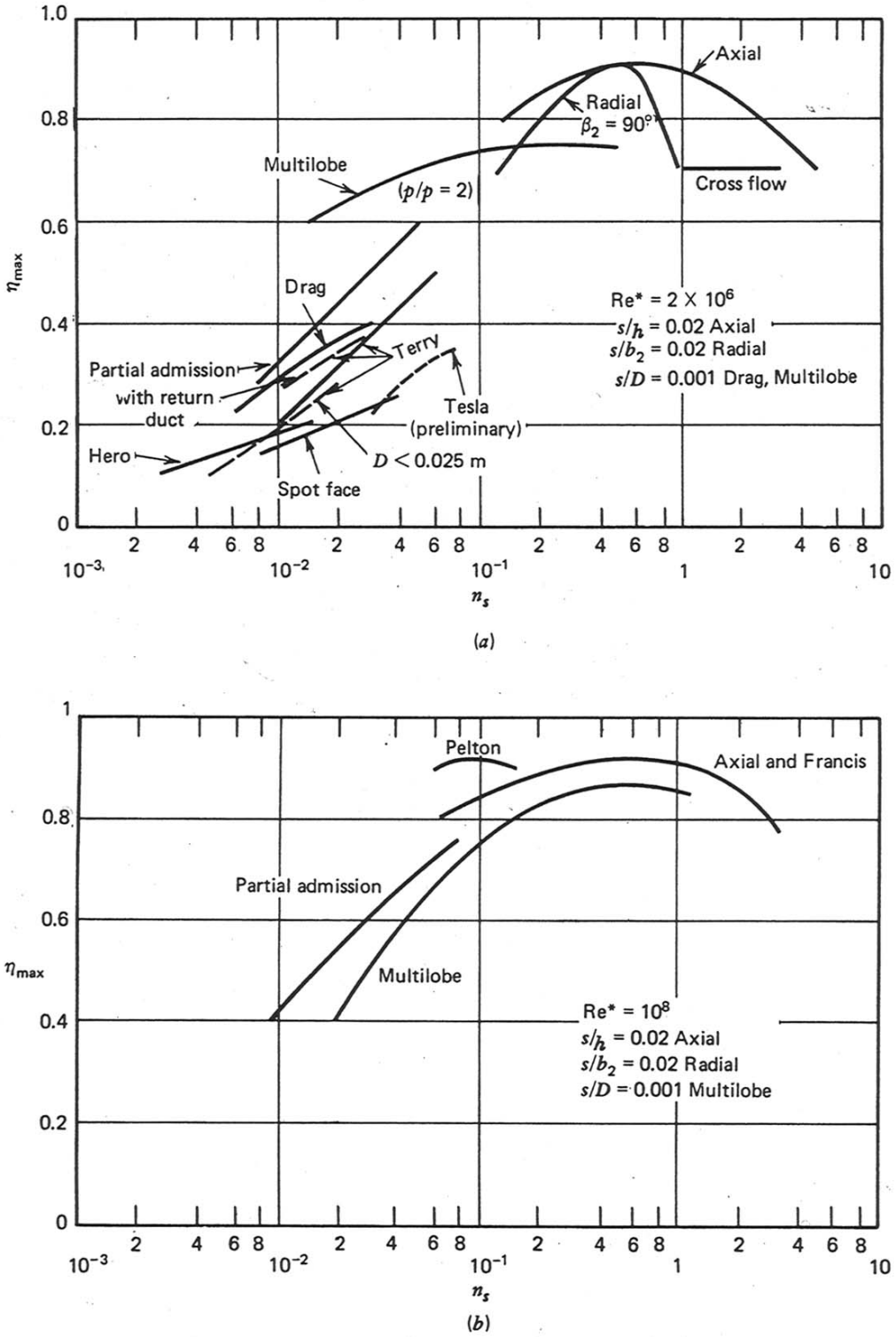


FIGURE 3.9 Maximum turbine efficiencies as function of specific speed.

Figure 14-10: Maximum Turbine Efficiencies (from [Balje])

In his text, Balje [Balje 1981, 54] states that for both radial compressors and radial turbines, the highest efficiencies are always achieved for designs with specific speeds between 0.50 and 1.00. A more recent Balje diagram, included in the text by [Logan 1993, 164], is reproduced here as Figure 14-11. It indicates that for purely radial inflow gas turbines the best efficiencies have been obtained with n_s=0.55.

14.5 Reynolds Number Effects

As noted in the plots of Figures 14-4 through 14-6, the presented efficiency design data for gas handling machines is appropriate for machine Reynolds numbers of 2 * 10^6, which is sufficiently high so that viscosity effects are not very important for efficiency. When considering machine applications depending on the triplet of dimensionless parameters (n_s, d_s, Re*), the achievable machine efficiency could theoretically be any function of those three parameters. However, all turbomachinery texts consulted presented arguments that the Reynolds number effects on efficiency are approximately separable. They claim achievable efficiency is approximately a product of an efficiency factor which is a function of Re* alone and another function of only (n_s, d_s), i.e., that

$$\eta \approx f_\eta(\mathrm{Re}^*) f(n_s, d_s) \tag{14-15}$$

Figure 14-12 reproduces Csanady's plots of how efficiency declines at lower Re* values. The upper plot also shows an analogous Re*-dependent factor to apply to head. (Note that Csanady's definition of Re* differs from Balje's definition by a factor of 2.) Balje [Balje 1981, 84, 86] also presents graphical data on how low machine Reynolds number hurts efficiency. Two of his graphs are reproduced here in Figure 14-13.

The Reynolds Number plots all agree that the efficiency factor for machine Reynolds numbers above 10^6 differ by only one or two percent, but that the efficiency factor declines to about 0.80 for machine Reynolds numbers of 10^5 and then plummets for even lower Reynolds numbers. A simple conclusion from inspecting these plots is that in spite of the fact that the kinematic viscosity of low pressure martian air is high due to its low density, the machine Reynolds number for the present design *must* be kept high. A value of Re* below 10^4 would likely be inadequate while Re* above 10^6 would almost certainly be adequate (if the turbomachine were otherwise well designed). Intermediate values of Re* would require careful analysis to see whether they could work at all within the tight energy budgets of a Brayton cycle thermal conversion system.

The kinematic viscosity of martian air at the nuclear engine's air intake is a given fact, not under the engine designer's control. The only way for the designer to increase the machine Reynolds number,

$$\mathrm{Re}^* \equiv \frac{ND^2}{2\nu} \equiv \frac{v_{tip} D}{\nu} \equiv \frac{\rho N D^2}{2\mu} \tag{14-16}$$

is to increase the ND^2 numerator. However, there are limits which constrain such increases. The rotor diameter, D, is limited by the physical size we are willing to use for the engine, and the rotor tip speed, ND/2, is limited by its effect on rotor stress.

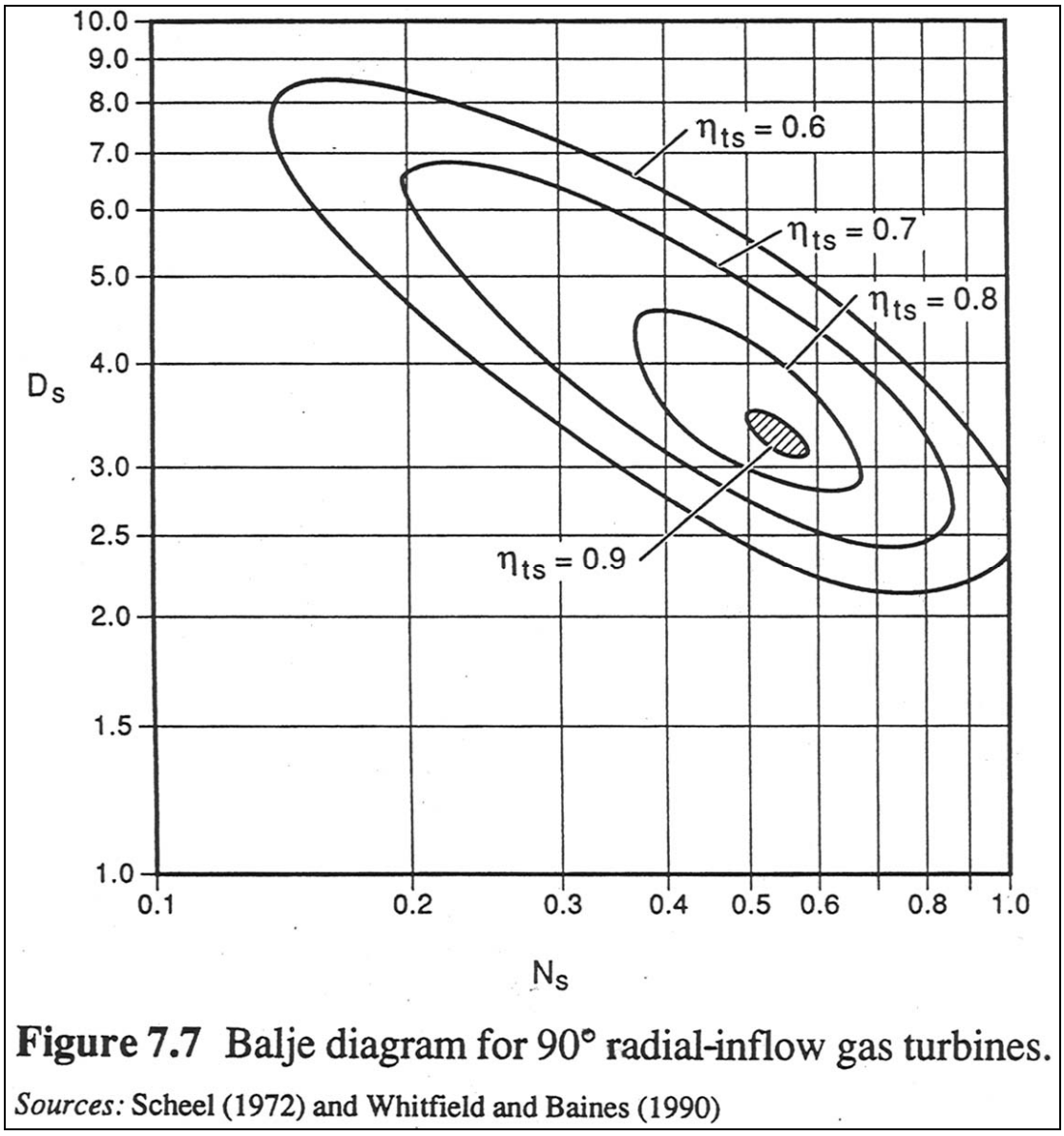


Figure 7.7 Balje diagram for 90° radial-inflow gas turbines.

Sources: Scheel (1972) and Whitfield and Baines (1990)

Figure 14-11: Turbine Efficiencies (from [Logan])

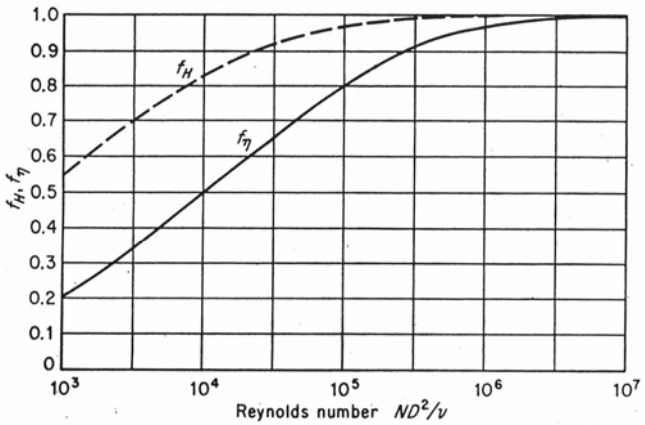


FIGURE 10. Reynolds-number factors for head and efficiency in centrifugal pumps of moderate size.

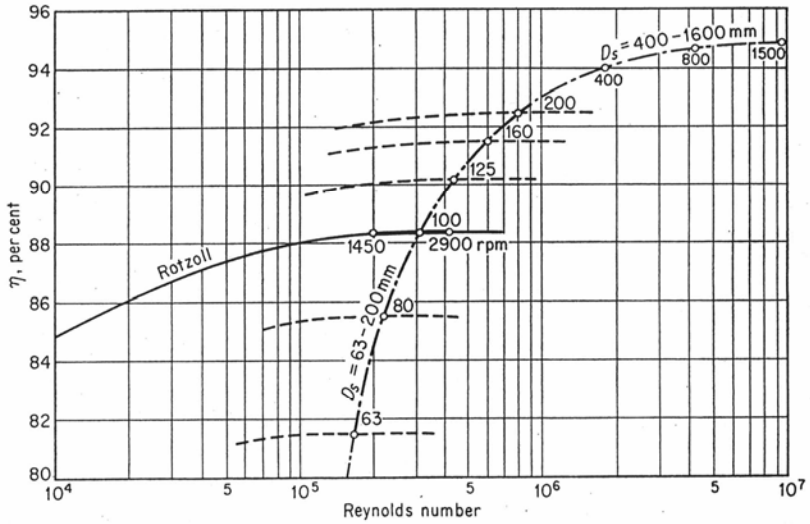


FIGURE 11. Efficiency versus Reynolds number for a series of similar machines (after Rutschi[19]).

Figure 14-12: Machine Reynolds Number Effects On Efficiency (from [Csanady])

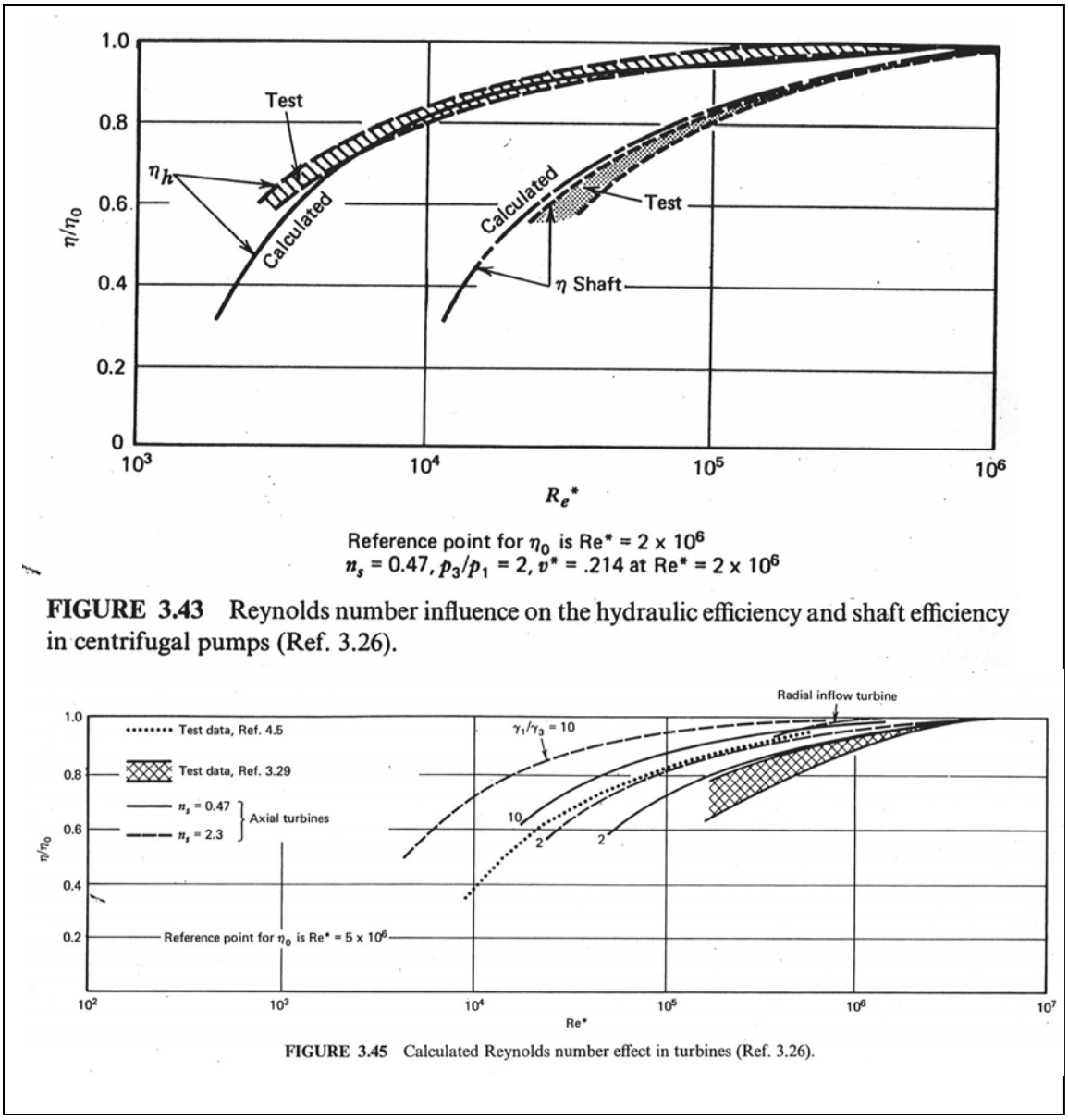


FIGURE 3.43 Reynolds number influence on the hydraulic efficiency and shaft efficiency in centrifugal pumps (Ref. 3.26).

FIGURE 3.45 Calculated Reynolds number effect in turbines (Ref. 3.26).

Figure 14-13: Machine Reynolds Number Efficiency Factors (from [Balje])

14.6 Stress Constraints On Rotor Tip Speed

Centrifugal force arising from rotation, also called apparent force or inertial force, causes most of the mechanical stress in a turbomachine's rotor. The rotor stress must be limited in order for the rotor to survive.

At a distance, r, from the rotor's axis of rotation, the apparent centrifugal force per unit volume of rotor material is directed radially outward from the rotation axis and is of magnitude $\rho N^2 r$, where ρ is the rotor material's mass density and N is the rotor's angular speed in radians per second. Written in vector form, the apparent body force per unit volume is as follows:

$$\vec{F}(\vec{r}) = \rho(\vec{r}) N^2 r \hat{r} \tag{14-17}$$

where the circumflex symbol above the final r denotes it is a unit vector oriented in the radial direction of a cylindrical coordinate system which is aligned with the rotation axis. For the rotor to remain intact, an equilibrium spatial pattern of internal material stresses within the rotor assembly must mechanically balance the centrifugal force pattern. This internal stress field is in general spatially nonuniform and it depends sensitively on the rotor's designed geometric shape. However, since all centrifugal forces causing these stresses are proportional to ρN^2, it logically follows that the resulting pattern of mechanical stresses within a rotor is also proportional to ρN^2.

14.6.1 Rotor Centrifugal Stress Scaling: Demonstrations And Proof

The internal stresses are actually proportional to $\rho N^2 R^2$, where R=D/2 is the radius of the rotor tip. Showing this is true is the purpose of the present section. To illustrate that the R^2 factor is also present, consider some idealized cases.

(a) Consider a slender rod of uniform cross section and length R, pinned on one end to a small axle in an orientation perpendicular to the axle. The axis rotates about its axis at N radians per second. Then at a location in the rod a distance of r from the axis the stress will be tensile in the radial direction with a magnitude as follows:

$$\sigma_r = \int_r^R \rho N^2 r dr = \frac{1}{2}\left(1 - \left(\frac{r}{R}\right)^2\right) \rho N^2 R^2 \tag{14-18}$$

Thus, stress in the rod is a product of a dimensionless function of size-independent, normalized position,

$$S\left(\frac{\vec{r}}{R}\right) = \frac{1}{2}\left(1 - \left(\frac{r}{R}\right)^2\right) \tag{14-19}$$

times $\rho N^2 R^2$. In this case the peak stress, $(\sigma_r)_{max} = S_{max}\ \rho N^2 R^2 = 0.5\ \rho N^2 R^2$, occurs at the connection with the axle where r/R=0 and S_{max}=0.50.

(b) Consider a circular hoop with cross section small compared to its radius, rotating about its geometric axis at N radians per second. Then the stress in the hoop will be uniform and purely tensile in the tangential direction (sometimes called the hoop direction or the azimuthal direction). The stress magnitude will be as follows:

$$\sigma_t = \rho N^2 R^2 \tag{14-20a}$$

Thus the stress in the hoop is the product of a constant function,

$$S\left(\frac{\vec{r}}{R}\right) = 1.0$$

times $\rho N^2 R^2$. (14-20b)

(c) Consider a homogeneous solid circular disk of outer radius, R, and of a uniform thickness which is thin compared to its radius, rotating about its axis at N radians per second. Then there will be nonzero tensile stress in both radial and tangential directions. The stress components will obey the following formulae, where $\nu_{Poisson}$ denotes the disk material's Poisson's number, a dimensionless elasticity property of materials which for most metals has values in the range from 0.25 to 0.35 [and Young 1975].

$$\sigma_r = \frac{1}{8}(3+\nu_{Poisson})\left(1-\left(\frac{r}{R}\right)^2\right)\rho N^2 D^2 \tag{14-21a}$$

$$\sigma_t = \frac{1}{8}\left((3+\nu_{Poisson})-(1+3\nu_{Poisson})\left(\frac{r}{R}\right)^2\right)\rho N^2 D^2 \tag{14-21b}$$

Thus, each stress component in a rotating disk is again the product of $\rho N^2 R^2$, with a dimensionless function of size-independent, normalized position, either

$$S_r\left(\frac{\vec{r}}{R}\right) = \frac{1}{8}(3+\nu_{Poisson})\left(1-\left(\frac{r}{R}\right)^2\right) \tag{14-22a}$$

or

$$S_t\left(\frac{\vec{r}}{R}\right) = \frac{1}{8}\left((3+\nu_{Poisson})-(1+3\nu_{Poisson})\left(\frac{r}{R}\right)^2\right) \tag{14-22b}$$

For the rotating disk the two stress components each reach identical maximum values at the disk's center.

(d) Consider a homogeneous annular disk of outer radius, R, inner radius, R_0, and of a uniform thickness which is thin compared to its radius, rotating about its axis at N radians per second. Then there will be nonzero tensile stress in both radial and tangential directions, as follows:

$$\sigma_r = \frac{1}{8}(3+\nu_{Poisson})\left(1+\left(\frac{R_0}{R}\right)^2-\left(\frac{r}{R}\right)^2-\left(\frac{R_0}{r}\right)^2\right)\rho N^2 D^2 \tag{14-23a}$$

$$\sigma_t = \frac{1}{8}\left((3+\nu_{Poisson})\left(1+\left(\frac{R_0}{R}\right)^2+\left(\frac{R_0}{r}\right)^2\right)-(1+3\nu_{Poisson})\left(\frac{r}{R}\right)^2\right)\rho N^2 D^2 \tag{14-23b}$$

Thus, in this case the material stress components are again the product of $\rho N^2 R^2$ with a dimensionless function of size-independent, normalized position, either

$$S_r\left(\frac{\vec{r}}{R}\right) = \frac{1}{8}(3+\nu_{Poisson})\left(1+\left(\frac{R_0}{R}\right)^2-\left(\frac{r}{R}\right)^2-\left(\frac{R_0}{r}\right)^2\right) \tag{14-24a}$$

or

$$S_t\left(\frac{\vec{r}}{R}\right) = \frac{1}{8}\left((3+\nu_{Poisson})\left(1+\left(\frac{R_0}{R}\right)^2+\left(\frac{R_0}{r}\right)^2\right)-(1+3\nu_{Poisson})\left(\frac{r}{R}\right)^2\right) \tag{14-24b}$$

The point of these examples is that spatial patterns of stress in rotating objects with geometrically similar shapes are themselves similar, that the effects of object size, rotation rate, and density can all be factored out, and that the proper factor is $\rho N^2 R^2$.

The idealized shapes considered above resemble portions of turbomachine rotors, but practical turbomachine rotor designs are far too complicated to obtain analytical solutions for their internal stresses resulting from their rotation. Finite element analysis (FEA) is usually conducted to numerically calculate the stress fields in rotor designs. However, those FEA computed stress fields can still be separated into a product of $\rho N^2 R^2$,with a dimensionless function of size-independent, normalized position which itself depends only on the rotor's normalized geometric shape. To see that this is true in all cases we examine the elasticity equations. The result follows without actually solving them, by application of the differentiation chain rule from calculus.

In the theory of elasticity as presented in various texts, the elastic *displacement* vector field,

$$\vec{u}(\vec{r}) \equiv \vec{r}^{\,(displaced)} - \vec{r} \equiv \begin{pmatrix} u_r(\vec{r}) \\ u_\phi(\vec{r}) \\ u_z(\vec{r}) \end{pmatrix}, \tag{14-25}$$

for an isotropic elastic solid obeys the following vector partial differential equation:

$$\nabla\nabla\bullet\vec{u} - \frac{1-2\nu_{Poisson}}{2(1-\nu_{Poisson})}\nabla\times\nabla\times\vec{u} \equiv -\frac{(1+\nu_{Poisson})(1-2\nu_{Poisson})}{E_{Young}(1-\nu_{Poisson})}\vec{F}(\vec{r}) \tag{14-26}$$

where E_{Young} is Young's modulus of elasticity for the material and *F* represents the body force per unit volume in the material. In this formulation of elasticity, thermal expansion effects are ignored. For a rotating body, the body force is expressed in cylindrical coordinates as

$$\vec{F}(\vec{r}) = \tilde{\rho}(\vec{r})N^2 r\hat{r} \tag{14-27}$$

Rotor surface boundary condition details can be avoided by defining the mass density as the product of a shape function times a constant, i.e.,

$$\tilde{\rho}(\vec{r}) = w(\vec{r})\rho \tag{14-28}$$

where w is defined as 1 wherever rotor material is present and 0 wherever it is not present. Following this approach, E_{Young} must also be reduced to a vanishingly small positive value wherever rotor material is absent.

When a displacement function solution to the elasticity equation is known, the six components of the associated symmetrical *strain tensor* can be calculated directly from it. In cylindrical coordinates they are found via the following relations:

$$u_{rr} = \frac{\partial u_r}{\partial r}; \qquad u_{\phi\phi} = \frac{1}{r}\frac{\partial u_\phi}{\partial \phi} + \frac{u_r}{r}; \qquad u_{zz} = \frac{\partial u_z}{\partial z}$$

$$2u_{\phi z} = \frac{1}{r}\frac{\partial u_z}{\partial \phi} + \frac{\partial u_\phi}{\partial z}; \qquad 2u_{rz} = \frac{\partial u_r}{\partial z} + \frac{\partial u_z}{\partial r};$$

$$2u_{r\phi} = \frac{\partial u_\phi}{\partial r} - \frac{u_\phi}{r} + \frac{1}{r}\frac{\partial u_r}{\partial \phi} \tag{14-29}$$

After the strain tensor is known, the *stress tensor* can be calculated from it without further differentiation, as follows.

$$\sigma_{rr} = \frac{E_{Young}}{(1+\nu_{Poisson})(1-2\nu_{Poisson})}\left[(1-\nu_{Poisson})u_{rr} + \nu_{Poisson}\left(u_{\phi\phi} + u_{zz}\right)\right]$$

$$\sigma_{\phi\phi} == \frac{E_{Young}}{(1+\nu_{Poisson})(1-2\nu_{Poisson})}\left[(1-\nu_{Poisson})u_{\phi\phi} + \nu_{Poisson}\left(u_{rr} + u_{zz}\right)\right] \tag{14-30}$$

$$\sigma_{zz} == \frac{E_{Young}}{(1+\nu_{Poisson})(1-2\nu_{Poisson})}\left[(1-\nu_{Poisson})u_{zz} + \nu_{Poisson}\left(u_{\phi\phi} + u_{rr}\right)\right]$$

$$\sigma_{\phi z} = \frac{E_{Young}}{(1+\nu_{Poisson})}u_{\phi z}; \qquad \sigma_{rz} = \frac{E_{Young}}{(1+\nu_{Poisson})}u_{rz}; \qquad \sigma_{r\phi} = \frac{E_{Young}}{(1+\nu_{Poisson})}u_{r\phi}$$

We assume that for a particular rotor with a tip radius of R_1 rotating at angular speed, N, with uniform material density $\rho^{(1)}$ and shape function, $w^{(1)}(\vec{r})$, that the particular displacement function which satisfies its elasticity equation is $\vec{u}^{(1)}(\vec{r})$. Stated differently, we assume that it satisfies the following:

$$\nabla\nabla \bullet \vec{u}^{(1)}(\vec{r}) - \frac{1-2\nu_{Poisson}}{2(1-\nu_{Poisson})}\nabla \times \nabla \times \vec{u}^{(1)}(\vec{r}) =$$

$$= -w^{(1)}(\vec{r})r\hat{r}\frac{(1+\nu_{Poisson})(1-2\nu_{Poisson})}{E_{Young}(1-\nu_{Poisson})}\rho N^2 \tag{14-31}$$

We then consider another rotor shape function, related to the first,

$$w^{(2)}(\vec{r}) = w^{(1)}\left(\frac{R_1}{R_2}\vec{r}\right) \equiv w^{(1)}(c\vec{r})$$

and will show that it is consistent with the following proposed displacement function:

$$\vec{u}^{(2)}(\vec{r}) = \left(\frac{R_2}{R_1}\right)^3 \vec{u}^{(1)}\left(\frac{R_1}{R_2}\vec{r}\right) \equiv \frac{\vec{u}^{(1)}(c\vec{r})}{c^3} \tag{14-32}$$

We denote the size ratio R_1/R_2 also by the symbol, c, simply to save notation. Note this second shape function is equivalent to photographically enlarging the rotor by a factor $c^{-1}=R_2/R_1$, so the second rotor must have a tip radius of R_2. The second rotor can be alternatively viewed as equivalent to defining a new coordinate system for the independent variables,

$$\vec{r}' = \frac{R_1}{R_2}\vec{r} \equiv c\vec{r} \tag{14-33}$$

so that the proposed solution becomes

$$\vec{u}^{(2)}(\vec{r}) = \left(\frac{R_2}{R_1}\right)^3 \vec{u}^{(1^3)}(\vec{r}') = \frac{\vec{u}^{(1)}(\vec{r}')}{c^3} \tag{14-34}$$

Then the matrix of partial derivatives connecting the two coordinate systems is an identity matrix times the dilatation constant, c, i.e. :

$$\frac{\partial \vec{r}'}{\partial \vec{r}} = c \tag{14-35}$$

If we define the primed symbol ∇' to denote the ∇ differentiation operation performed with respect to the $\vec{r}'$ primed independent variables, then the chain rule for partial differentiation becomes the following operator equation:

$$c\nabla' \equiv \nabla \tag{14-36}$$

Since the derivatives on the left hand side of Eq. 14-31 are second order, two factors of the constant, c, result from applying the chain rule to it. Indeed, substituting

$$\vec{u}^{(2)}(\vec{r}) = \frac{\vec{u}^{(1)}(\vec{r}')}{c^3} \tag{14-37}$$

into the left hand side yields the following reduction.

$$\begin{aligned}
&\nabla\nabla \bullet \vec{u}^{(2)}(\vec{r}) - \frac{1-2\nu_{Poisson}}{2(1-\nu_{Poisson})}\nabla \times \nabla \times \vec{u}^{(2)}(\vec{r}) = \\
&= \nabla\nabla \bullet \frac{\vec{u}^{(1)}(\vec{r}')}{c^3} - \frac{1-2\nu_{Poisson}}{2(1-\nu_{Poisson})}\nabla \times \nabla \times \frac{\vec{u}^{(1)}(\vec{r}')}{c^3} = \\
&= \frac{1}{c}\left[\nabla'\nabla' \bullet \vec{u}^{(1)}(\vec{r}') - \frac{1-2\nu_{Poisson}}{2(1-\nu_{Poisson})}\nabla' \times \nabla' \times \vec{u}^{(1)}(\vec{r}')\right]
\end{aligned} \tag{14-38}$$

In Eq. 14-38 the final bracketed quantity matches the left hand side of Eq. 14-31 but with the primed independent variables. Therefore the right hand side of Eq. 14-31 can replace it providing that primed independent variables are used. Continuing the reduction,

$$\frac{1}{c}\left[\nabla'\nabla'\bullet\vec{u}^{(1)}(\vec{r}')-\frac{1-2\nu_{Poisson}}{2(1-\nu_{Poisson})}\nabla'\times\nabla'\times\vec{u}^{(1)}(\vec{r}')\right]=$$

$$=\frac{1}{c}\left[-w^{(1)}(\vec{r}')r'\hat{r}\frac{(1+\nu_{Poisson})(1-2\nu_{Poisson})}{E_{Young}(1-\nu_{Poisson})}\rho N^2\right] \tag{14-39}$$

$$=\frac{1}{c}\left[-w^{(1)}(c\vec{r})cr\hat{r}\frac{(1+\nu_{Poisson})(1-2\nu_{Poisson})}{E_{Young}(1-\nu_{Poisson})}\rho N^2\right]$$

$$=\left[-w^{(2)}(\vec{r})r\hat{r}\frac{(1+\nu_{Poisson})(1-2\nu_{Poisson})}{E_{Young}(1-\nu_{Poisson})}\rho N^2\right]$$

Ignoring all the intermediate steps in 14-38 and 14-39 then yields the following:

$$\nabla\nabla\bullet\vec{u}^{(2)}(\vec{r})-\frac{1-2\nu_{Poisson}}{2(1-\nu_{Poisson})}\nabla\times\nabla\times\vec{u}^{(2)}(\vec{r})=$$

$$=-w^{(2)}(\vec{r})r\hat{r}\frac{(1+\nu_{Poisson})(1-2\nu_{Poisson})}{E_{Young}(1-\nu_{Poisson})}\rho N^2 \tag{14-40}$$

But Eq. 14-40 is again the elasticity equation, so this ansatz has shown that the proposed second displacement function does in fact satisfy the elasticity equation if the first does.

Strain tensor components are then determined from this displacement function via the partial derivatives of Eqs. 14-29. The chain rule is again applied here by taking the derivatives with respect to the primed independent variables then multiplying results by *c*, which cancels one of the *c* factors in the denominator of Eq. 14-32b. Stress tensor components are then found via Eqs 14-30, which have no derivatives. The result is:

$$\sigma_{ij}{}^{(2)}(\vec{r}')=\frac{\sigma_{ij}{}^{(1)}(\vec{r})}{c^2}\equiv\frac{R_2{}^2}{R_1{}^2}\sigma_{ij}{}^{(1)}(\vec{r}) \tag{14-41}$$

or equivalently

$$\frac{\sigma_{ij}{}^{(2)}(\vec{r}')}{R_2{}^2}=\frac{\sigma_{ij}{}^{(1)}(\vec{r})}{R_1{}^2} \tag{14-42}$$

Since it was previously established that mechanical stress is proportional to ρN^2, Eq. 14-42 completes the proof that mechanical stress at corresponding locations scale in proportion to $\rho N^2 R^2$.

14.6.2 Rotor Stress Limit Effects Of Temperature And Time

The product *NR* is the tangential speed of the rotor tip due to its rotation. Thus, this stress scaling result states that stress components anywhere in a rotor are the product of S, a

dimensionless function of normalized position in the rotor, with rotor density and the square of the rotor tip speed.

$$\sigma_{ij}(\vec{r}') = S_{ij}(\vec{r}')\rho v_{tip}^2 \tag{14-43}$$

The maximum allowable rotor tip speed occurs with rotor stress equal to its material's allowable stress. It is calculated as follows:

$$v_{tip} = \sqrt{\frac{1}{S_{\max}}}\sqrt{\frac{\sigma_{\max}}{\rho}} \tag{14-44}$$

Balje's example for typical rotors of radial flow turbomachines is reproduced here as Figure 14-14. [Balje 1981, 61] Its upper cross section shows a tapered rotor disk (grey) and an attached vane blade (clear) rotating about the vertical axis on the left. Maximum normalized stress is about 0.2 for this design and occurs at about 80% of the rotor's tip radius.

Since for efficient turbomachine operation at a high pressure ratio per stage it is important that rotor tip speed be fast, this ratio of allowable stress to mass density is a crucially important material property for turbomachine design. However, the allowable stress is itself a function of material temperature, a fact which complicates the design process. Balje includes on his page 30 plots showing this ratio versus temperature for material types that were popular at the time of his textbook's publication. It is regrettable that Balje represented allowable stress in weight-kg/m^2 units, i.e., using the kilogram as a unit of force instead of using the modern SI unit of pressure, the Pascal (i.e., one Newton of force per square meter). Today, stresses are usually stated in MPa (MegaPascals). The numerical values labeling his vertical axis must therefore be multiplied by g=9.8 m/sec^2, the acceleration of gravity at Earth's surface, in order to calculate actual values of σ/ρ in their proper m^2/sec^2 units for use in Eq. 14-49.

For most candidate metallic rotor materials the failure phenomena which limits maximum allowable stress is creep. The stressed material slowly flows in a growing plastic deformation that eventually ends in a material rupture. The material's plastic flow rate depends on its level of stress and on its temperature. Therefore the maximum allowable stress in the rotor material depends not only on the material's temperature, but also on how much time we require the rotor to operate before it fails. As shown, the Figure 14-15 data is for 1000 hours to rupture failure. It would not be valid for a lifetime of, say, 10,000 hours or 100,000 hours. Balje provided a plot showing the joint dependence of allowable stress on both temperature and the time to failure for Inconel 713, a popular nickel alloy at the time of his text's publication. It is reproduced below as Figure 14-16. Regrettably, it also uses pressure stress units of weight-kg/m^2 which must be multiplied by g=9.8 m/sec^2 to obtain the material stress in Pascals.

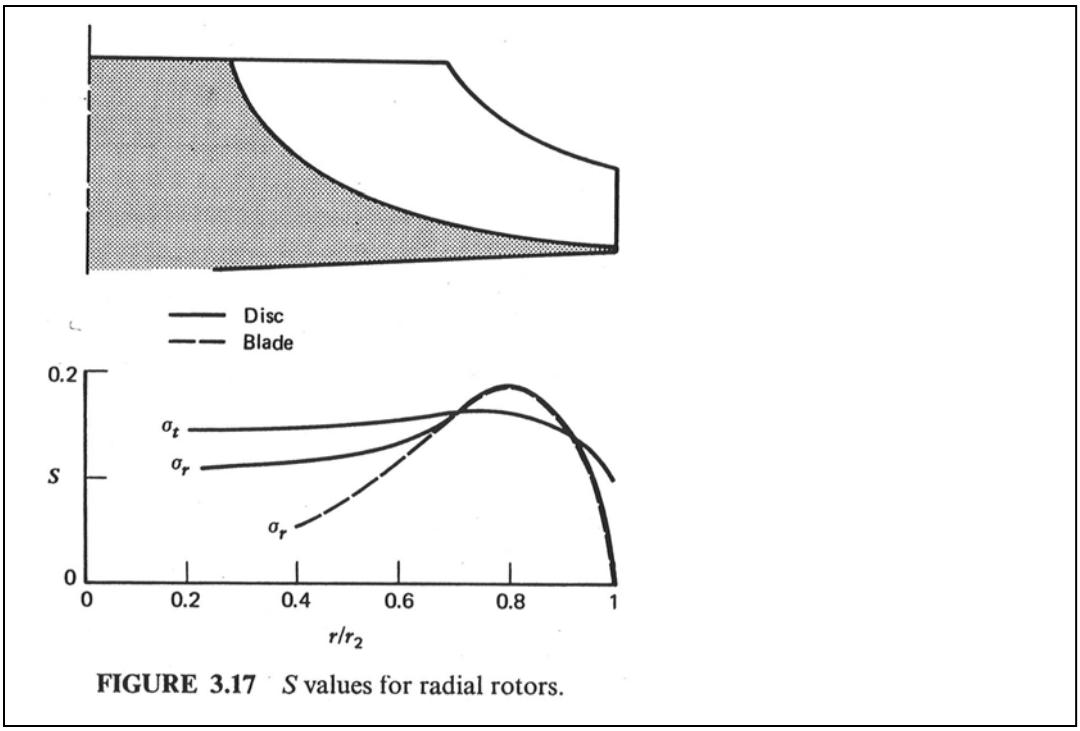


FIGURE 3.17 *S* values for radial rotors.

Figure 14-14: Typical Stress Factors In Radial Flow Turbomachine Rotors (from [Balje])

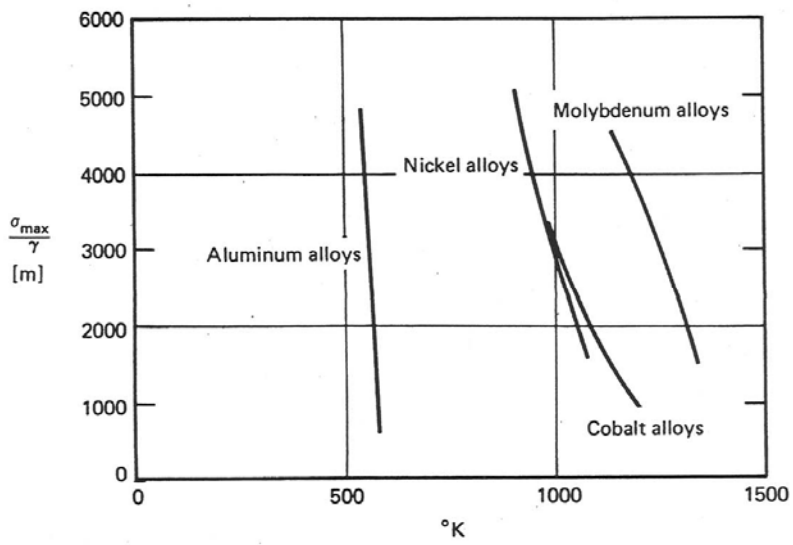


FIGURE 2.7 Maximum allowable stresses for different materials for 1000 hr rupture life.

Figure 14-15: Property Limits Vs Temperature For Candidate Rotor Materials (from [Balje])

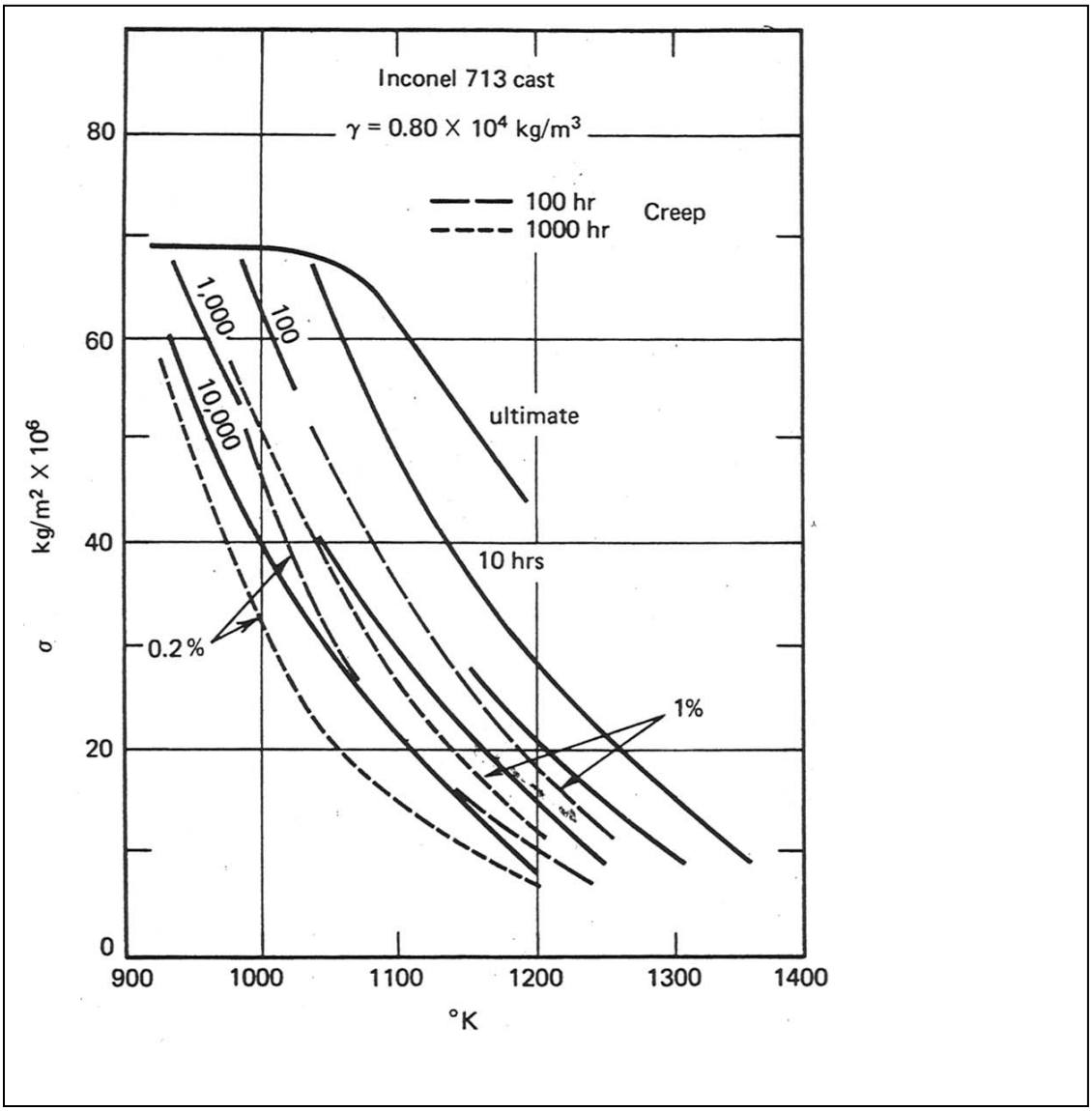


Figure 14-16: Allowable Stress Vs Temperature And Time To Failure Of Cast Inconel 713 Alloy (from [Balje])

Development of new turbomachine rotor materials with either better mechanical properties or lower cost has been a well financed quest of applied research for the past several decades as aircraft jet engines and fixed site gas power systems have become increasingly important to the global economy. Thus, Balje's illustrative use of Inconel 713 may be obsolete. Checking with a materials engineer, Marc Leuzzi [telephone interview June 2006] who is familiar with the modern turbomachine industry, reveals that today Inconel 718 and Inconel 738 have become the popular materials for commercial turbomachine rotors of the radial flow type. An alloy supplier, Haynes, provides on their internet web site, http://www.haynesintl.com , a comparison of the 1000 hour rupture temperature-dependent creep strength of their 718 alloy product with several other candidate alloys which they also sell; their plot is reproduced here as Figure 14-17.

Perhaps one of these may be best here, or perhaps instead a more expensive exotic material may be superior for a Mars application where material cost is utterly negligible compared to the cost of transportation to Mars. As shown in Figure 14-15, molybdenum alloys (e.g., TZM) can operate with similar creep behavior at about 200 Kelvins higher temperatures than precipitation hardened nickel alloys (i.e., inconel).

It should be understood that the temperature dependence of allowable stress is not an issue for all turbomachines. In systems using multiple compressor and turbine stages, only the highest pressure *turbine* is subjected to the highest temperature. The highest pressure compressor may produce a somewhat high temperature in its outlet gas flow but that temperature is not as high as in the turbine. Heat addition boosts the temperature of gas from the compressor before the gas enters the high pressure turbine, either via combustion as in the usual terrestrial Brayton cycle, or via a heat exchanger as in the present nuclear engine design. Lower pressure stages operate at cooler temperatures.

Temperatures within the high pressure turbine's rotor are never uniform since the gas it contacts has an extremely nonuniform temperature. Within a radial inflow high pressure turbine the highest temperature gas is always located at the rotor tip where the gas enters. The extraction of work from the gas by the turbine cools the gas remarkably as the gas flows inwards, so the gas is much cooler as it exits near the turbine's axis. Thus, material at the rotor tip location reaches the highest temperature. On the other hand, mechanical stress is low or zero at the rotor tip as shown in Figure 14-14, so rotor tip conditions are not limiting. The limiting combination of stress and temperature always occurs somewhere between the rotor tip and the maximum stress location, which may be at the 0.8R radial location if the rotor design is similar to Figure 14-14. To determine this limitation today requires a complicated computer-based analysis of rotor stress, gas flow and gas temperature vs position near the rotor, heat transfer between the external gas and the rotor, and heat conduction within the rotor. Decades ago this limitation would have been explored by extensive model experiments and their associated diagnostic measurements. Either approach would exceed the resources available for the present design effort, so the design developed here will rely on published past experience.

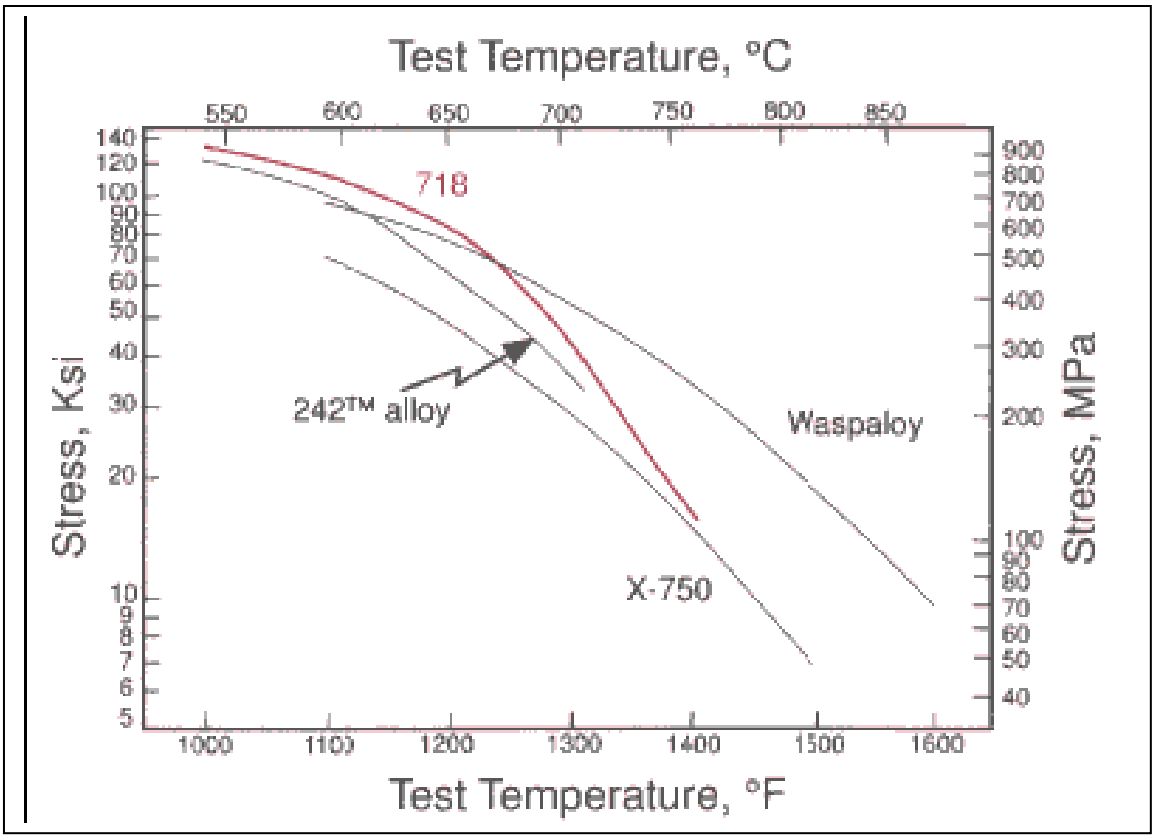


Figure 14-17: Creep Stress Vs. Temperature For 1000 hour Rupture Life In Alloy 718 And Other Candidate Turbine Rotor Materials (from Haynes internet web site, http://www.haynesintl.com/)

A feature of some contemporary turbine designs is active cooling of the rotor. This can reduce the temperature at the rotor's critical locations where creep effects constrain allowable stress. Thus, active cooling can allow a higher rotor tip speed than would be feasible without it. The active cooling is provided by diverting a small part of the pressurized air emerging from the compressor to directly flow through internal cooling passages inside the rotor blades without first receiving the heat addition which is provided to the main flow of gas in the Brayton cycle. This cooling gas stream first flows into, then through, axial passages in the turbine's rotating shaft, then it turns outward to flow through additional cooling passages inside the turbine's disc and vane blades before exiting through tiny holes into the main flow stream. A designer must decide whether the complexity needed for such active cooling is warranted in relation to the performance benefit it can provide. Not all turbine designs use active cooling of their rotors.

It is common to replace turbine rotors after operating them for their rated number of hours in order to avoid rupture failures. For a nuclear engine operating on Mars, it may instead be prudent to design its turbomachine rotors for a very long or even an infinite life without failing due to creep. It is likely possible to do this since it will not be feasible for the nuclear engine's heat exchanger to heat martian air to reach temperatures quite as hot as those routinely developed in the flames feeding most combustion turbines. It is also possible to choose exotic metals capable of higher temperature operation for the high pressure turbine rotor, such as the molybdenum alloys suggested by Figure 14-15.

14.6.3 Expected Rotor Tip Speeds

Converting the data presented in Figure 14-19 to modern SI units, the maximum resulting temperature-dependent values for the different alloy curves shown, i.e., values for cold operation, are between $4*10^4$ m^2 s^{-2} and $5*10^4$ m^2 s^{-2}. Thus, for these alloys, the quantity

$$\sqrt{\frac{\sigma_{max}}{\rho}}$$

is within the range of speeds from 200 m/s to 224 m/s. For other alloy stress data presented in Figures 14-20 and 14-21, first dividing stress by alloy density then performing similar calculations results in about the same numerical range of speeds.

The maximum allowed rotor tip speed for these alloys (operating cold) then depends on the geometrical shape of the rotor design as it affects the maximum value of its normalized stress, S_{max}. For the most limiting shape, i.e., a circular hoop as in case (b), rotating about its symmetry axis, the normalized stress is everywhere $S=S_{max}=1.0$. Therefore, the tip speed for rotors shaped like hoops would be constrained according to Eq. 14-44 to not exceed 200 to 224 m/s.

Most shapes permit higher speeds. For individual rotor blades approximating a rod rooted to a central shaft of very small radius as in case (a), the maximum normalized stress is $S_{max}=0.50$. The maximum tip speed allowed for these alloys by Eq. 14-44 is then about 283 to 316 m/s. Higher speeds would be allowed if their rooting shaft is not small.

For a solid cylindrical disk of uniform axial thickness made from material with a Poisson's ratio of 1/3, the maximum normalized stress is S_{max}=0.417. The maximum tip speed for such a disk constructed from these alloys allowed by Eq. 14-44 is then about 310 to 347 m/s.

Axially tapered disks allow much higher tip speeds. For the particular specially designed tapered rotor disk with vane blades rooted to it as shown in Figure 14-14, the stated maximum normalized stress is S_{max}=0.200. The maximum tip speed for it constructed from these alloys as allowed by Eq. 14-44 is then about 450 to 500 m/s. It is worthwhile to notice that this high allowable tip speed is obtained through a special shaping of the rotor which has evolved over years of successive designs. It requires a specially tapered solid rotor disk for structural support which is far more massive than the attached vane blades interacting directly with the air.

Even higher rotor tip speeds are possible with a comparatively recent innovation (see [Balje]) in which the rotor disk is scalloped in a non-axisymmetric fashion between vane blades, giving it local bulges where the blades are attached to it. The higher tip speed capability results because such shaping further lowers maximum normalized stress.

This range of maximum operating rotor tip speed values is confirmed by typical values mentioned in turbomachine textbooks. For example, Logan states [Logan 1993, 165] that "1600-1700 fps is a possible range of acceptable values for tip speed" for a radial inflow 90 degree IFR type of turbine. Converted to modern SI units this speed range becomes 488 m/s to 518 m/s. The much older turbomachine text by Shepherd, [Shepherd 1956, 450] stated that 450 m/s was (then) typical but that 500 m/s was "also possible".

It is not essential that very high tip speeds be used. If lower tip speeds are adopted in order to limit stress, the lower resulting pressure ratios per stage could be accommodated by using more stages in a multistage design. The cost impact of limiting maximum rotor tip speeds is thus its effect on the thermal conversion system's complexity and mass. Using known rotor materials and making only small changes to existing turbine designs, the rotor tip speed for turbomachines in nuclear engines for Mars will not be any higher than present typical rotor tip speeds for terrestrial applications. Indeed, they may even be deliberately set slightly lower in order to obtain longer rotor life. It seems reasonable to assume rotor tip speed should be in the range from 450 to 500 m/s.

14.7 Rotor Diameter

Turbomachine rotor diameters for Open Brayton Cycle applications on Mars will be larger than is common on Earth. There are two reasons for this. First, a large diameter is necessary to avoid a low machine Reynolds number and its consequent lossy turbomachine performance. Second, a large size is needed at the air intake in order to

draw in a sufficient mass flow of the martian air. Both of these reasons apply mainly to the low pressure stages. The higher pressure stages can be physically smaller.

The machine Reynolds number definition of Eq. 14-16 can solved for rotor diameter:

$$D = \frac{2\,\mathrm{Re}^*}{v_{tip}}\nu \qquad (14\text{-}45)$$

If rotor tip speed is set to v_{tip}=450 m/sec and the machine Reynolds number to Re*=2*10^6 in order to match Balje's diagrams , the Figure 14-18 contour plot results, presenting the minimum rotor diameters needed to avoid any reduction in turbomachine efficiency due to the viscous losses which can plague low pressure gas systems.

Thus, by selecting a rotor diameter in the 3 meter to 4 meter range, the largest that seems feasible for a practical mobile engine, the low pressure compressor can be designed to operate at very high efficiencies for most of the ambient air intake conditions the mobile engine will encounter anywhere on Mars' surface. The portion of Figure 14-18's ambient air rectangle for which such high efficiency operation is not assured is the lower right corner, which corresponds to simultaneous low pressure and high temperature. That condition is uncommon since such low pressures will only be experienced at high altitude martian surface locations where the martian air temperature is almost never warm.

However, with the low pressure turbine limited to the same 3 to 4 meters diameter as the low pressure compressor, it will certainly suffer some efficiency penalty. The low pressure turbine's exhaust pressure will be close to the ambient atmospheric pressure but the exhaust temperature will be considerably hotter. For instance, as shown in Figure 14-18, if the exhaust temperature were 600 Kelvins then at some seasons and high altitude locations for which the air pressure may drop to 350 Pa, a turbine rotor diameter around 40 meters would be needed for fully efficient operation without low Reynolds number viscous losses reducing efficiency. With a 4 meter diameter rotor the low pressure turbine's machine Reynolds number for those conditions would be reduced to 2*10^5; the corresponding turbine efficiency change according to the graphical data of Figures 14-12 and 14-13 would multiply the high Reynolds number efficiency by between 0.8 and 0.9.

For the low pressure turbine, the design strategy must include acceptance of some efficiency reduction from the ideal, but with attempts to minimize that reduction. As shown in Figure 14-18, if the exhaust temperature at martian atmospheric pressure were reduced then the necessary rotor diameter for high efficiency turbine operation would also be reduced. This implies that low pressure turbine operation with a rotor diameter between 3 and 4 meters would also become more efficient. In the open Brayton cycle that design point shift would result from increasing the compressor's compression pressure ratio to increase cycle efficiency. Thus, this logic provides a reason to depart from the theoretically maximum specific work design point as developed in the previous chapter, in the direction of increased efficiency. That chapter did not consider efficiency dependencies on *absolute* cycle pressures, e.g., via such machine Reynolds number effects.

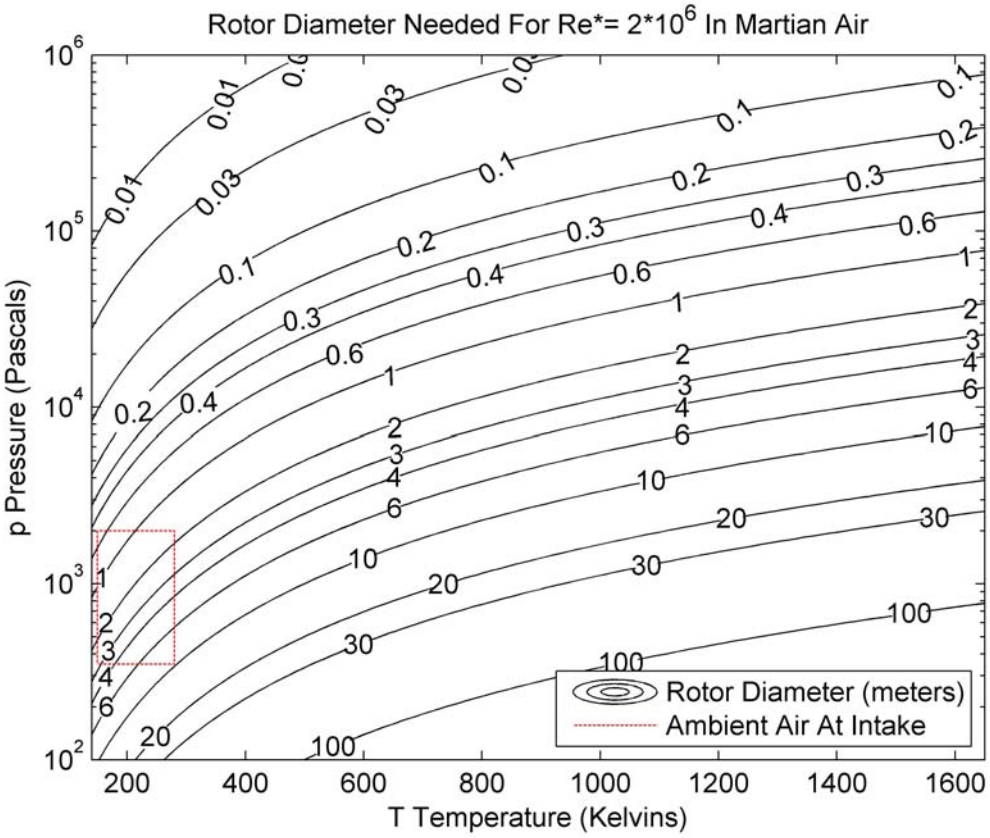


Figure 14-18: Turbomachine Rotor Diameter Required For Very High Efficiency Operation With Martian Air (i.e., For v_{tip}=450 m/s & Re*=2*10^6)

In principle, all of the higher pressure compressor and turbine stages could be designed to operate with diameters sufficiently large to be free from Reynolds number effects. However, as summarized by the Balje diagrams of Figures 14-12 and 14-15, the achievable turbomachine efficiency is also a strong function of specific speed and specific diameter, (N_s,D_s), each of which partly depends on Q, the low pressure side's volumetric flowrate. Choosing a large rotor diameter for small flow rates could thus sabotage efficiency even though it would eliminate Reynolds number effects.

14.8 Choosing Compressor And Turbine Stage Parameters

In this section, results are summarized of sizing calculations documented in Appendix D for the nine engine models of Table 4-1 (ranging from 100 to 10,000 horsepower).

The graphical performance data excerpted from texts and included here as Figures 14-8 through 14-11 indicate the ranges of dimensionless parameters which in existing designs have maximized efficiencies of turbomachines. It should be recognized that these performance correlations were compiled based on the full rated design conditions and do not reflect operation at part-load conditions. Figure 14-8(b) indicates that for radial flow compressors the best efficiencies have followed from choosing Ns=0.7, Ds=3.7, which is selected here for design of efficient radial flow compressors. For radial flow turbines, Figure 14-8(a) indicates the optimum has been Ns=0.55, Ds=3.37, while Figure 14-11 indicates the optimum has been Ns=0.55, Ds=3.25. Roughly averaging them, Ns=0.55, Ds=3.3 is selected for design of efficient radial inflow turbines. Ideally, each compressor and turbine in a multistage design should be assigned these ideal dimensionless parameters. Based on cumulative industry experience, that would guarantee a detailed conventional design of each would develop an efficiency among the highest exhibited by past and present turbomachines. However, interrelations and practical limits may compromise this ideal.

In the Brayton cycle, a considerable portion of the power developed in turbine stages must be fed back to operate compressor stages. The usual way to achieve this power feedback is to simply mount compressor and turbine stages on a common shaft which mechanically transmits torque between them as they rotate together at the same rotational speed. This was the only practical way to efficiently transfer power between turbomachine stages until about 2 decades ago, when progress in power electronics provided the alternative of efficient chopper-based variable speed drives for brushless induction motor/generator devices. These have subsequently been deployed globally for railroad wheel drives and are now being deployed for other traction applications, e.g., in hybrid automobiles. Such systems can efficiently transfer large amounts of power in either direction between shafts rotating at unrelated, arbitrary, and even changing speeds.

In spite of the new design freedom provided by modern power electronics, mounting compressors and turbines on common shafts remains the power feedback method requiring the least equipment, so it is used here to the extent practical. However, with the large pressure ratios developed by each radial flow turbomachine stage, successive stages are poorly matched to rotate at the same speed. The impact of forcing successive stages to rotate at the same angular speed instead of at their most efficient design speeds could be high. Therefore, to best avoid incurring a large performance penalty, each shaft couples compressors with turbines operating over similar fluid pressure ranges. Power electronics can then still accommodate a power mismatch between them by controlling and varying the relative amounts of power extracted from the different shafts. Transient imbalances with the external load can also be accommodated through power electronics by transferring the balancing power from or to an attached electrical battery system.

The Figures 14-19 and 14-20 block diagrams show configurations implemented in this fashion with two and three shafts, respectively. It should be recognized that other configurations are also possible. An alternative configuration not shown here would include one more turbine stage than compressor stages and would mount the highest pressure turbine stage on its own shaft with no corresponding compressor stage.

Mounting compressor and turbine stages on the same shaft reduces the number of free parameters which the designer may assign, since the co-rotating turbomachines don't have independent angular rotation rates. With co-rotation it is not possible in general to obtain dimensionless parameters independently maximizing efficiency of each stage. The best that can be done is a compromise.

Initial exploratory calculations using MATLAB were conducted assuming the intake air was at 215 Kelvins and 850 Pa and taking the maximum air temperature after adding heat from the nuclear reactor to be 1290 Kelvins. For these bounding assumptions, the analyses of Chapter 13 had previously found the optimum pressure ratio for maximizing specific work, and thus minimizing air flow for fixed output power, to be 132.57 assuming ideally perfect components as summarized in Table 13-1. In that analysis the maximum air pressure in the cycle, occurring at the heat exchanger inlet, was 112,685 Pa, which happens to be 111.2% of Earth's atmospheric pressure at sea level. As shown in Figures 13-27 and 13-30, the pressure ratio to maximize specific work drops to the range of 25 to 30 for turbomachine isentropic efficiencies of 0.80 and heat exchanger pressure ratios between 0.80 and 1.00. It was found that if turbomachine rotor speeds are kept below 500 m/s and the high temperature turbine's rotor tip speed is further restricted to about 450 m/s, then the three shaft configuration is necessary to obtain pressure ratios above 100 but the two shaft configuration is sufficient for pressure ratios of 25 to 30.

As discussed in Chapter 13, maximum efficiency designs, which minimize reactor power and thus minimize radiation shielding mass for fixed output power levels, require a

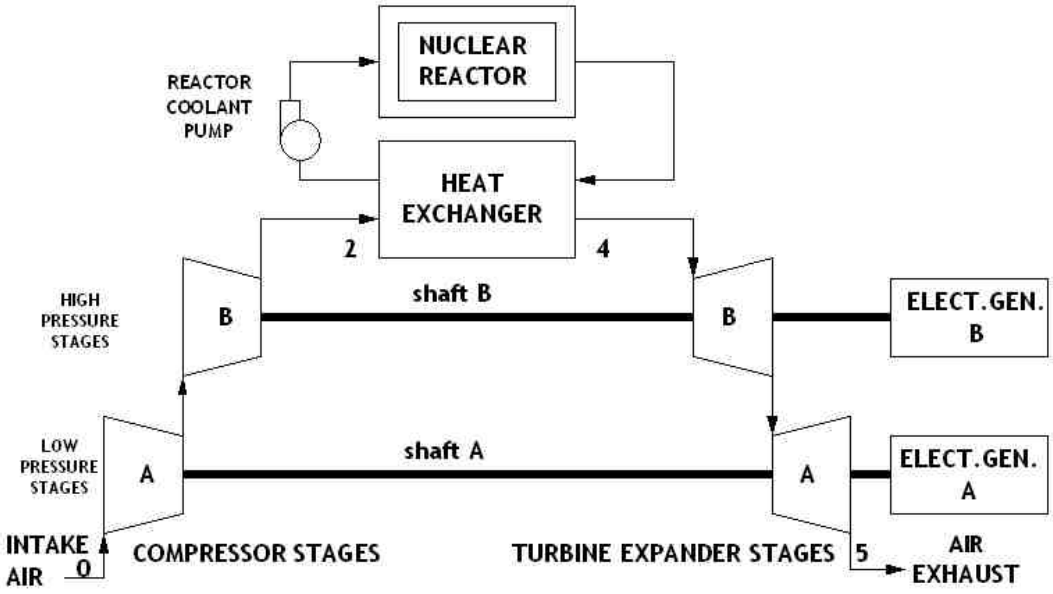


Figure 14-19: A Multi-Stage Open Brayton Cycle Using Two Shafts

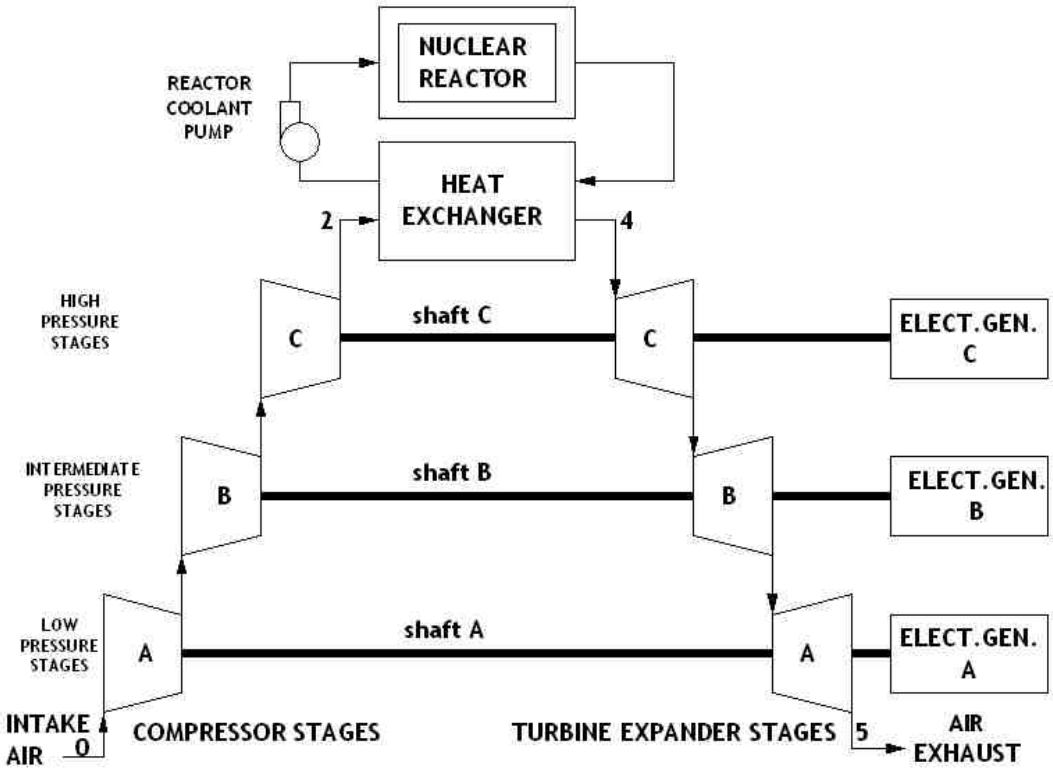


Figure 14-20: A Multi-Stage Open Brayton Cycle Using Three Shafts

higher pressure ratio than the maximum specific work designs which minimize airflow. Since the true optimum design point is likely to be somewhere between the two, it was decided to adopt the three-shaft configuration. MATLAB script (see Appendix E) was written to calculate turbomachine parameters using actual MarsMix gas thermodynamic properties, based on input values of reversible work per unit mass for each stage, on Balje diagram evaluation of an efficiency factor based on (n_s,d_s) and on machine Reynolds number efficiency factors calculated by implementing the upper curve in Figure 14-12 as a MATLAB interpolation table. The input reversible work values were first adjusted to obtain acceptable calculated rotor tip speeds, then were left fixed throughout the subsequent calculations for different engine power levels.

For the nine designs of Table 4-1, the resulting calculated design parameters are summarized below in Tables 14-2 through 14-10. Of those, the six lower power engine designs, with output power levels ranging from 100 horsepower to 1,780 horsepower, strictly follow the 3-shaft scheme shown in Figure 14-20. Since machines used are of the radial flow type, each of the three shafts is most conveniently mounted in a vertical orientation with their attached turbomachine rotors and the attached motor-generator's rotor rotating in the horizontal plane. Figure 14-21 depicts this configuration in a cross-section view. The low pressure turbine and compressor are sufficiently large that they must be mounted as a canopy at an elevation well above the other engine components including the reactor, shielding, heat exchanger, and the higher pressure turbomachine stages. Within the low pressure shaft assembly, the low pressure turbine is mounted on the top so that its hot exhaust can be expelled vertically upwards as it emerges from the turbine's eye. The compressor below it draws in its intake air axially upwards into its own eye after the air has first flowed in the horizontal plane radially inwards from all sides. To least impact these external airflows, the electrical motor-generator is mounted between the turbine and the compressor on their common shaft.

The higher pressure turbomachines which are smaller but geometrically similar are implemented in the same fashion. However, the gas flows to and from their eyes are entirely enclosed within ducts connected to the volutes of lower pressure turbomachine stages. None of their eyes is open to the low pressure external air. The heat exchanger is connected between the volutes of the high pressure compressor and turbine stages.

In the three highest engine power designs, ranging from 3,160 horsepower to 10,000 horsepower, each low pressure stage is implemented as multiple turbomachines operating in parallel in order to avoid rotor diameters in excess of 4 meters. The numbers of machines operating in parallel in these very high power designs are all divisible by 2, so these are implemented most effectively by using dual-sided radial flow turbomachines such as the unit shown in Figure 14-2 for each parallel pair. The low pressure turbines are all located together on the same shaft, vertically above all of the low pressure compressor stages. Higher pressure stages are implemented as individual stages in a manner unchanged from the lower power engine designs.

Table 14-2: Engine Model #1 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**100 horsepower**					
Output Power (SI units)	**74.6 kW**					
Thermal Input Power (from Nuclear Reactor)	**310.0 kW**					
Thermal Conversion Cycle Energy Efficiency	**0.2406**					
Air Mass Flow Rate (kg/sec)	**0.4194 kg/sec**					
	Low Pressure Stages		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**979.8 radians/sec**		**2,365.2 radians/sec**		**4,072.8 radians/sec**	
Shaft Speed (rpm)	**9,356.6 rpm**		**22,586.4 rpm**		**38,892.8 rpm**	
	Compressor	**Turbine**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**0.912 m**	**1.006 m**	**0.374 m**	**0.417 m**	**0.219 m**	**0.223 m**
Rotor Tip Speed (m/s)	**446.7 m/s**	**492.9 m/s**	**442.0 m/s**	**492.9 m/s**	**445.0 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.07**	**3.86**	**3.00**	**3.84**	**3.05**
Ns: Specific Speed	**0.656**	**0.590**	**0.642**	**0.605**	**0.651**	**0.595**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$7.68*10^5$**	**$1.10*10^5$**	**$1.06*10^6$**	**$2.18*10^5$**	**$1.38*10^6$**	**$3.71*10^5$**
Reynolds Number Efficiency Factor	**0.957**	**0.810**	**0.971**	**0.878**	**0.976**	**0.923**
Assumed Efficiency (from Balje*Reynolds)	**0.813**	**0.712**	**0.825**	**0.772**	**0.829**	**0.812**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**942.49 Kelvins**	**399.1 Kelvins**	**1128.8 Kelvins**	**555.6 Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,211 Pa**	**9,592 Pa**	**33,924 Pa**	**38,504 Pa**	**102,011 Pa**
Pressure Ratio (Calculated)	**11.285**	**6.851**	**4.014**	**4.705**	**2.944**	**3.007**
Discharge Air Temperature (Kelvins)	**399.1 Kelvins**	**762.7 Kelvins**	**555.6 Kelvins**	**942.4 Kelvins**	**698.5 Kelvins**	**1128.8 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**1,053 Pa**	**38,504 Pa**	**7,211 Pa**	**113,345 Pa**	**33,924 Pa**
Q:Volumetric Flow Rate On Low Pressure Side	**20.27 m^3/s**	**58.07 m^3/s**	**3.33 m^3/s**	**10.47 m^3/s**	**1.16 m^3/s**	**2.67 m^3/s**
AirSpeed Through Eye(1)	**97.0 m/s**	**228.3 m/s**	**94.7 m/s**	**239.6 m/s**	**96.2 m/s**	**213.6 m/s**
Local Sound Speed at Eye	**234.2 m/s**	**418.4 m/s**	**310.1 m/s**	**462.9 m/s**	**360.8 m/s**	**504.8 m/s**
Shaft Powers (kW) [Total=+74.6 kW=100 hp]	**-65.48 kW**	**+88.15 kW**	**-64.53 kW**	**+95.561kW**	**-64.23 kW**	**+85.14 kW**

(1) Eye Annulus extends from 20% to 60% of Rotor Tip Radius

Table 14-3: Engine Model #2 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**178 horsepower**					
Output Power (SI units)	**132.8 kW**					
Thermal Input Power (from Nuclear Reactor)	**495.8 kW**					
Thermal Conversion Cycle Energy Efficiency	**0.2648**					
Air Mass Flow Rate (kg/sec)	**0.6684 kg/sec**					
	Low Pressure Stages		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**777.1 radians/sec**		**1,882.0 radians/sec**		**3,243.3 radians/sec**	
Shaft Speed (rpm)	**7,420.7 rpm**		**17,972.0 rpm**		**30,970.8 rpm**	
	Compressor	**Turbine**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**1.150 m**	**1.269 m**	**0.470 m**	**0.524**	**0.274 m**	**0.280 m**
Rotor Tip Speed (m/s)	**446.9 m/s**	**492.9 m/s**	**442.4 m/s**	**492.9 m/s**	**445.1 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.08**	**3.86**	**3.01**	**3.84**	**3.05**
Ns: Specific Speed	**0.657**	**0.589**	**0.643**	**0.603**	**0.651**	**0.595**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$9.61*10^5$**	**$1.41*10^5$**	**$1.35*10^6$**	**$2.79*10^5$**	**$1.75*10^6$**	**$4.73*10^5$**
Reynolds Number Efficiency Factor	**0.968**	**0.835**	**0.975**	**0.903**	**0.980**	**0.934**
Assumed Efficiency (from Balje*Reynolds)	**0.823**	**0.734**	**0.829**	**0.794**	**0.833**	**0.822**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**934.9 Kelvins**	**397.2 Kelvins**	**1126.7 Kelvins**	**555.6 Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,255 Pa**	**9,592 Pa**	**34,245 Pa**	**553.2Kelvins**	**102,974 Pa**
Pressure Ratio (Calculated)	**11.285**	**6.981**	**4.037**	**4.720**	**38,727 Pa**	**3.007**
Discharge Air Temperature (Kelvins)	**397.2 Kelvins**	**749.1 Kelvins**	**553.2Kelvins**	**934.9 Kelvins**	**695.7 Kelvins**	**1126.7 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**1,039 Pa**	**38,727 Pa**	**7,255 Pa**	**114,415 Pa**	**34,245 Pa**
Q:Volumetric Flow Rate On Low Pressure Side	**32.30 m^3/s**	**92.06 m^3/s**	**5.29 m^3/s**	**16.46 m^3/s**	**1.82 m^3/s**	**4.20 m^3/s**
AirSpeed Through Eye(1)	**97.2 m/s**	**227.5 m/s**	**95.3 m/s**	**238.5 m/s**	**96.5 m/s**	**213.2 m/s**
Local Sound Speed at Eye	**234.2 m/s**	**414.9 m/s**	**309.5 m/s**	**461.1 m/s**	**360.8 m/s**	**504.4 m/s**
Shaft Powers (kW) [Total=132.8 kW=178 hp]	**-103.16 kW**	**+144.82 kW**	**-102.41 kW**	**+156.60 kW**	**-101.93 kW**	**+137.37 kW**

(1) Eye Annulus extends from 20% to 60% of Rotor Tip Radius

Table 14-4: Engine Model #3 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**316 horsepower**					
Output Power (SI units)	**235.7 kW**					
Thermal Input Power (from Nuclear Reactor)	**821.7 kW**					
Thermal Conversion Cycle Energy Efficiency	**0.2869**					
Air Mass Flow Rate (kg/sec)	**1.1054 kg/sec**					
	Low Pressure Stages		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**604.6 radians/sec**		**1,467.3 radians/sec**		**2,529.9 radians/sec**	
Shaft Speed (rpm)	**5774.0 rpm**		**14,011.3 rpm**		**24,159.2 rpm**	
	Compressor	**Turbine**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**1.479 m**	**1.630 m**	**0.603 m**	**0.672**	**0.352 m**	**0.359 m**
Rotor Tip Speed (m/s)	**447.1 m/s**	**492.9 m/s**	**442.7 m/s**	**492.9 m/s**	**445.2 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.08**	**3.86**	**3.01**	**3.84**	**3.05**
Ns: Specific Speed	**0.657**	**0.589**	**0.644**	**0.602**	**0.651**	**0.594**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$1.24*10^6$**	**$2.27*10^5$**	**$1.74*10^6$**	**$3.63*10^5$**	**$2.27*10^6$**	**$6.11*10^5$**
Reynolds Number Efficiency Factor	**0.974**	**0.861**	**0.980**	**0.922**	**0.984**	**0.946**
Assumed Efficiency (from Balje*Reynolds)	**0.828**	**0.758**	**0.833**	**0.811**	**0.837**	**0.833**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**928.5 Kelvins**	**396.2 Kelvins**	**1124.6 Kelvins**	**551.7Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,267 Pa**	**9,592 Pa**	**34,424 Pa**	**38,839 Pa**	**103,513Pa**
Pressure Ratio (Calculated)	**11.285**	**7.096**	**4.049**	**4.737**	**2.961**	**3.007**
Discharge Air Temperature (Kelvins)	**396.2 Kelvins**	**736.2 Kelvins**	**551.7Kelvins**	**928.5 Kelvins**	**693.6 Kelvins**	**1124.6 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**1,024 Pa**	**38,839 Pa**	**7,267 Pa**	**115,014 Pa**	**34,424 Pa**
Q:Volumetric Flow Rate On Low Pressure Side	**53.42 m^3/s**	**151.83 m^3/s**	**8.72 m^3/s**	**26.99 m^3/s**	**3.00 m^3/s**	**6.90 m^3/s**
AirSpeed Through Eye(1)	**97.2 m/s**	**227.4 m/s**	**95.3 m/s**	**237.8 m/s**	**96.3 m/s**	**213.0 m/s**
Local Sound Speed at Eye	**234.2 m/s**	**411.5 m/s**	**309.1 m/s**	**459.6 m/s**	**359.6 m/s**	**503.9 m/s**
Shaft Powers (kW) [Total=235.7 kW=316 hp]	**-169.63 kW**	**+247.15 kW**	**-168.60 kW**	**+264.46 kW**	**-167.82 kW**	**+230.18 kW**

(1) Eye Annulus extends from 20% to 60% of Rotor Tip Radius

Table 14-5: Engine Model #4 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**562 horsepower**					
Output Power (SI units)	**419.3 kW**					
Thermal Input Power (from Nuclear Reactor)	**1.368 MW**					
Thermal Conversion Cycle Energy Efficiency	**0.3065**					
Air Mass Flow Rate (kg/sec)	**1.837 kg/sec**					
	Low Pressure Stages		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**469.5 radians/sec**		**1,140.4 radians/sec**		**1,968.0 radians/sec**	
Shaft Speed (rpm)	**4,482,9 rpm**		**10,890.3 rpm**		**18,792.8 rpm**	
	Compressor	**Turbine**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**1.905 m**	**2.100 m**	**0.777 m**	**0.864**	**0.453 m**	**0.461 m**
Rotor Tip Speed (m/s)	**447.3 m/s**	**492.9 m/s**	**442.9 m/s**	**492.9 m/s**	**445.3 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.09**	**3.86**	**3.02**	**3.83**	**3.05**
Ns: Specific Speed	**0.657**	**0.588**	**0.645**	**0.601**	**0.652**	**0.594**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$1.59*10^6$**	**$2.41*10^5$**	**$2.25*10^6$**	**$4.71*10^5$**	**$2.93*10^6$**	**$7.92*10^5$**
Reynolds Number Efficiency Factor	**0.978**	**0.888**	**0.984**	**0.934**	**0.989**	**0.959**
Assumed Efficiency (from Balje*Reynolds)	**0.831**	**0.781**	**0.836**	**0.822**	**0.840**	**0.844**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**923.5 Kelvins**	**395.5 Kelvins**	**1122.3 Kelvins**	**550.3 Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,272 Pa**	**9,592 Pa**	**34,574 Pa**	**38,927 Pa**	**103,962 Pa**
Pressure Ratio (Calculated)	**11.285**	**7.188**	**4.058**	**4.754**	**2.967**	**3.007**
Discharge Air Temperature (Kelvins)	**395.5 Kelvins**	**724.7 Kelvins**	**550.3 Kelvins**	**923.5 Kelvins**	**691.8 Kelvins**	**1122.3 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**1,012 Pa**	**38,927 Pa**	**7,272 Pa**	**115,513 Pa**	**34,574 Pa**
Q:Volumetric Flow Rate On Low Pressure Side	**88.77 m^3/s**	**251.40 m^3/s**	**14.47 m^3/s**	**44.57 m^3/s**	**4.96 m^3/s**	**11.39 m^3/s**
AirSpeed Through Eye(1)	**97.3 m/s**	**226.8 m/s**	**95.4 m/s**	**237.6 m/s**	**96.2 m/s**	**213.2 m/s**
Local Sound Speed at Eye	**234.2 m/s**	**408.4 m/s**	**308.9 m/s**	**458.4 m/s**	**359.1 m/s**	**503.4 m/s**
Shaft Powers (kW) [Total=419.3 kW=562 hp]	**-280.60 kW**	**+423.44 kW**	**-278.89 kW**	**+445.41 kW**	**-277.59 kW**	**+387.48 kW**

(1) Eye Annulus extends from 20% to 60% of Rotor Tip Radius

Table 14-6: Engine Model #5 Turbocompressor Parameters At Full Design Point Operationn

Output Power (US customary units)	**1000 horsepower**					
Output Power (SI units)	**746.0 kW**					
Thermal Input Power (from Nuclear Reactor)	**2.291 MW**					
Thermal Conversion Cycle Energy Efficiency	**0.3256**					
Air Mass Flow Rate (kg/sec)	**3.071 kg/sec**					
	Low Pressure Stages		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**363.4 radians/sec**		**883.8 radians/sec**		**1,526.2 radians/sec**	
Shaft Speed (rpm)	**3,470.7 rpm**		**8,439.4 rpm**		**13,574.6 rpm**	
	Compressor	**Turbine**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**2.462 m**	**2.712 m**	**1.003 m**	**1.115**	**0.584 m**	**0.595 m**
Rotor Tip Speed (m/s)	**447.5 m/s**	**492.9 m/s**	**443.2 m/s**	**492.9 m/s**	**445.3 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.09**	**3.85**	**3.02**	**3.83**	**3.06**
Ns: Specific Speed	**0.658**	**0.587**	**0.645**	**0.600**	**0.652**	**0.594**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$2.06*10^6$**	**$3.17*10^5$**	**$2.91*10^6$**	**$6.14*10^5$**	**$3.81*10^6$**	**$1.03*10^6$**
Reynolds Number Efficiency Factor	**0.983**	**0.915**	**0.989**	**0.946**	**0.992**	**0.970**
Assumed Efficiency (from Balje*Reynolds)	**0.835**	**0.805**	**0.840**	**0.833**	**0.843**	**0.854**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**918.6 Kelvins**	**394.7 Kelvins**	**1120.3 Kelvins**	**549.0Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,279 Pa**	**9,592 Pa**	**34,724 Pa**	**39,016 Pa**	**104,416 Pa**
Pressure Ratio (Calculated)	**11.285**	**7.281**	**4.067**	**4.770**	**2.974**	**3.007**
Discharge Air Temperature (Kelvins)	**394.7 Kelvins**	**713.2 Kelvins**	**549.0Kelvins**	**918.6 Kelvins**	**690.1 Kelvins**	**1120.3 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**1,000 Pa**	**39,016 Pa**	**7,279 Pa**	**116,018 Pa**	**34,724 Pa**
Q:Volumetric Flow Rate On Low Pressure Side	**148.39 m^3/s**	**418.51 m^3/s**	**24.14 m^3/s**	**74.94 m^3/s**	**8.25 m^3/s**	**18.93 m^3/s**
AirSpeed Through Eye[(1)]	**97.4 m/s**	**226.4 m/s**	**95.5 m/s**	**239.8 m/s**	**96.2 m/s**	**212.8 m/s**
Local Sound Speed at Eye	**234.2 m/s**	**405.4 m/s**	**308.6 m/s**	**457.2 m/s**	**358.8 m/s**	**503.0 m/s**
Shaft Powers (kW) [Total=746 kW=1000 hp]	**-466.93 kW**	**+729.42 kW**	**-464.08 kW**	**+754.65 kW**	**-462.66 kW**	**+655.59 kW**

(1) Eye Annulus extends from 20% to 60% of Rotor Tip Radius

Table 14-7: Engine Model #6 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**1780 horsepower**					
Output Power (SI units)	**1.328 MW**					
Thermal Input Power (from Nuclear Reactor)	**3.931 MW**					
Thermal Conversion Cycle Energy Efficiency	**0.3378**					
Air Mass Flow Rate (kg/sec)	**5.260 kg/sec**					
	Low Pressure Stages		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**277.9 radians/sec**		**676.6 radians/sec**		**1168.8 radians/sec**	
Shaft Speed (rpm)	**2,653.4 rpm**		**6,461.4 rpm**		**11,161.3 rpm**	
	Compressor	**Turbine**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**3.222 m**	**3.548 m**	**1.311 m**	**1.457 m**	**0.762 m**	**0.776 m**
Rotor Tip Speed (m/s)	**447.6 m/s**	**492.9 m/s**	**443.4 m/s**	**492.9 m/s**	**445.4 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.81**	**3.09**	**3.85**	**3.03**	**3.83**	**3.06**
Ns: Specific Speed	**0.659**	**0.587**	**0.646**	**0.600**	**0.652**	**0.594**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$2.70*10^6$**	**$4.19*10^5$**	**$3.82*10^6$**	**$8.10*10^5$**	**$5.00*10^6$**	**$1.35*10^6$**
Reynolds Number Efficiency Factor	**0.987**	**0.928**	**0.992**	**0.960**	**0.994**	**0.975**
Assumed Efficiency (from Balje*Reynolds)	**0.839**	**0.817**	**0.843**	**0.845**	**0.845**	**0.858**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**914.9 Kelvins**	**394.0 Kelvins**	**1119.5 Kelvins**	**547.8 Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,299 Pa**	**9,592 Pa**	**34,870 Pa**	**39,109 Pa**	**104,852 Pa**
Pressure Ratio (Calculated)	**11.285**	**7.354**	**4.077**	**4.777**	**2.979**	**3.007**
Discharge Air Temperature (Kelvins)	**394.0 Kelvins**	**706.0 Kelvins**	**547.8 Kelvins**	**914.9 Kelvins**	**688.7 Kelvins**	**1119.5Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**993 Pa**	**39,109 Pa**	**7,299 Pa**	**116,502 Pa**	**34,870 Pa**
Q:Volumetric Flow Rate On Low Pressure Side	**254.22 m^3/s**	**714.92 m^3/s**	**41.28 m^3/s**	**125.97 m^3/s**	**14.08 m^3/s**	**32.27 m^3/s**
AirSpeed Through Eye(1)	**97.4 m/s**	**226.0 m/s**	**95.6 m/s**	**236.1 m/s**	**96.5 m/s**	**213.2 m/s**
Local Sound Speed at Eye	**234.2 m/s**	**403.4 m/s**	**308.3 m/s**	**450.3 m/s**	**358.4 m/s**	**502.8 m/s**
Shaft Powers (kW) [Total=1.328MW=1780hp]	**-0.796 MW**	**+1.268 MW**	**-0.793 MW**	**+1.311 MW**	**-0.791 MW**	**+1.129 MW**

(1) Eye Annulus extends from 20% to 60% of Rotor Tip Radius

Table 14-8: Engine Model #7 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**3160 horsepower**					
Output Power (SI units)	**2.357 MW**					
Thermal Input Power (from Nuclear Reactor)	**6.879 MW**					
Thermal Conversion Cycle Energy Efficiency	**0.3427**					
Air Mass Flow Rate (kg/sec)	**9.203 kg/sec**					
	Low Pressure Stages Operating In Parallel		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**296.6 radians/sec**		**511.7radians/sec**		**883.7 radians/sec**	
Shaft Speed (rpm)	**2,832.6 rpm**		**4,886.6 rpm**		**8,438.6 rpm**	
	Compressors (2)	**Turbines (2)**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**3.016 m**	**3.323 m**	**1.733 m**	**1.926 m**	**1.008 m**	**1.027 m**
Rotor Tip Speed (m/s)	**447.3 m/s**	**492.9 m/s**	**443.5 m/s**	**492.9 m/s**	**445.4 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.09**	**3.85**	**3.03**	**3.84**	**3.06**
Ns: Specific Speed	**0.658**	**0.588**	**0.647**	**0.600**	**0.652**	**0.594**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$2.52*10^6$**	**$3.91*10^5$**	**$5.06*10^6$**	**$1.08*10^6$**	**$6.62*10^6$**	**$1.79*10^6$**
Reynolds Number Efficiency Factor	**0.986**	**0.925**	**0.994**	**0.971**	**0.996**	**0.980**
Assumed Efficiency (from Balje*Reynolds)	**0.838**	**0.814**	**0.845**	**0.855**	**0.847**	**0.862**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**911.5 Kelvins**	**394.2 Kelvins**	**1118.6 Kelvins**	**547.6 Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,287 Pa**	**9,592 Pa**	**34,859 Pa**	**39,086 Pa**	**104,820 Pa**
Pressure Ratio (Calculated)	**11.285**	**7.420**	**4.075**	**4.784**	**2.980**	**3.007**
Discharge Air Temperature (Kelvins)	**394.2 Kelvins**	**703.2 Kelvins**	**547.6 Kelvins**	**911.5 Kelvins**	**688.2 Kelvins**	**1118.6 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**982 Pa**	**39,086 Pa**	**7,287 Pa**	**116,467 Pa**	**34,859 Pa**
Q:Volumetric Flow Rate Total Low Pressure Sides	**444.77 m^3/s**	**1,259.20 m^3/s**	**72.25 m^3/s**	**219.96 m^3/s**	**24.64 m^3/s**	**56.43 m^3/s**
AirSpeed Through Eye(1)	**97.3 m/s**	**226.9 m/s**	**95.7 m/s**	**235.9` m/s**	**96.5 m/s**	**212.9 m/s**
Local Sound Speed at Eye	**234.2 m/s**	**402.7 m/s**	**308.4 m/s**	**455.5 m/s**	**358.4 m/s**	**502.6 m/s**
Shaft Powers (kW) [Total=2.357MW=3160hp]	**-1.395 MW**	**+2.210 MW**	**-1.383 MW**	**+2.321 MW**	**-1.380MW**	**+1.984 MW**

(1) Eye Annulus extends from 20% to 60% of Rotor Tip Radius

Table 14-9: Engine Model #8 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**5620 horsepower**					
Output Power (SI units)	**4.193 MW**					
Thermal Input Power (from Nuclear Reactor)	**12.136 MW**					
Thermal Conversion Cycle Energy Efficiency	**0.3455**					
Air Mass Flow Rate (kg/sec)	**16.232 kg/sec**					
	Low Pressure Stages Operating In Parallel		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**315.5 radians/sec**		**385.3 radians/sec**		**665.4 radians/sec**	
Shaft Speed (rpm)	**3,012.7 rpm**		**3,679.4 rpm**		**6,354.5 rpm**	
	Compressors (4)	**Turbines (4)**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**2.834 m**	**3.125 m**	**2.302 m**	**2.558 m**	**1.339 m**	**1.364 m**
Rotor Tip Speed (m/s)	**447.0 m/s**	**492.9 m/s**	**443.6 m/s**	**492.9 m/s**	**445.4 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.08**	**3.85**	**3.03**	**3.84**	**3.06**
Ns: Specific Speed	**0.657**	**0.589**	**0.647**	**0.599**	**0.652**	**0.594**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$2.37*10^6$**	**$3.66*10^5$**	**$6.71*10^6$**	**$1.43*10^6$**	**$8.79*10^6$**	**$2.38*10^6$**
Reynolds Number Efficiency Factor	**0.985**	**0.922**	**0.997**	**0.976**	**0.999**	**0.985**
Assumed Efficiency (from Balje*Reynolds)	**0.837**	**0.811**	**0.847**	**0.859**	**0.849**	**0.867**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**909.5 Kelvins**	**394.3 Kelvins**	**1117.7 Kelvins**	**547.4 Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,274 Pa**	**9,592 Pa**	**34,850 Pa**	**39,064 Pa**	**104,795 Pa**
Pressure Ratio (Calculated)	**11.285**	**7.460**	**4.072**	**4.791**	**2.981**	**3.007**
Discharge Air Temperature (Kelvins)	**394.3 Kelvins**	**701.8 Kelvins**	**547.4 Kelvins**	**909.5 Kelvins**	**687.7 Kelvins**	**1117.7 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**975 Pa**	**39,064 Pa**	**7,274 Pa**	**116,438 Pa**	**34,850 Pa**
Q:Volumetric Flow Rate Total Low Pressure Sides	**784.46 m^3/s**	**2232.16 m^3/s**	**127.49 m^3/s**	**387.76 m^3/s**	**43.36 m^3/s**	**99.47 m^3/s**
AirSpeed Through Eye(1)	**97.2 m/s**	**227.4 m/s**	**95.7 m/s**	**235.8` m/s**	**96.2 m/s**	**212.7 m/s**
Local Sound Speed at Eye	**234.2 m/s**	**402.3 m/s**	**308.4 m/s**	**455.0 m/s**	**358.3 m/s**	**502.4 m/s**
Shaft Powers (kW) [Total=4.193MW=5,620hp]	**-2.462 MW**	**+3.885 MW**	**-2.434 MW**	**+4.114 MW**	**-2.428 MW**	**+3.518 MW**

(1) Eye Annulus extends from 20% to 60% of Rotor Tip Radius

Table 14-10: Engine Model #9 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**10,000 horsepower**					
Output Power (SI units)	**7.460 MW**					
Thermal Input Power (from Nuclear Reactor)	**21.274 MW**					
Thermal Conversion Cycle Energy Efficiency	**0.3507**					
Air Mass Flow Rate (kg/sec)	**28.432 kg/sec**					
	Low Pressure Stages Operating In Parallel		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**291.6 radians/sec**		**291.6 radians/sec**		**503.3 radians/sec**	
Shaft Speed (rpm)	**2,784.7 rpm**		**2,784.7 rpm**		**4,806.2 rpm**	
	Compressors (6)	**Turbines (6)**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**3.064 m**	**3.380 m**	**3.044 m**	**3.380 m**	**1.770 m**	**1.803 m**
Rotor Tip Speed (m/s)	**446.7 m/s**	**492.9 m/s**	**443.9 m/s**	**492.9 m/s**	**445.5 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.08**	**3.85**	**3.03**	**3.84**	**3.06**
Ns: Specific Speed	**0.656**	**0.588**	**0.648**	**0.600**	**0.652**	**0.594**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$2.56*10^6$**	**$3.97*10^5$**	**$8.89*10^6$**	**$1.90*10^6$**	**$1.17*10^7$**	**$3.15*10^6$**
Reynolds Number Efficiency Factor	**0.986**	**0.926**	**0.999**	**0.981**	**1.000**	**0.990**
Assumed Efficiency (from Balje*Reynolds)	**0.838**	**0.815**	**0.849**	**0.863**	**0.850**	**0.871**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**907.5Kelvins**	**394.1 Kelvins**	**1116.9 Kelvins**	**546.9 Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,275 Pa**	**9,592 Pa**	**34,904 Pa**	**39,091 Pa**	**104,957 Pa**
Pressure Ratio (Calculated)	**11.285**	**7.460**	**4.072**	**4.791**	**2.981**	**3.007**
Discharge Air Temperature (Kelvins)	**394.1 Kelvins**	**698.8 Kelvins**	**546.9 Kelvins**	**907.5Kelvins**	**687.0 Kelvins**	**1116.9 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**970 Pa**	**39,091 Pa**	**7,275 Pa**	**116,619 Pa**	**34,904 Pa**
Q:Volumetric Flow Rate Total Low Pressure Sides	**1,374.04 m^3/s**	**3913.40 m^3/s**	**223.19 m^3/s**	**677.64 m^3/s**	**76.00 m^3/s**	**173.82 m^3/s**
AirSpeed Through Eye(1)	**97.0 m/s**	**227.2 m/s**	**95.8 m/s**	**236.0` m/s**	**96.5 m/s**	**212.7 m/s**
Local Sound Speed at Eye	**234.2 m/s**	**401.5 m/s**	**308.4 m/s**	**454.6 m/s**	**358.1 m/s**	**502.3 m/s**
Shaft Powers (kW) [Total=7.460MW=10,000hp]	**-4.307 MW**	**+6.834 MW**	**-4.252 MW**	**+7.242 MW**	**-4.248 MW**	**+6.192 MW**

(1) Eye Annulus extends from 20% to 60% of Rotor Tip Radius

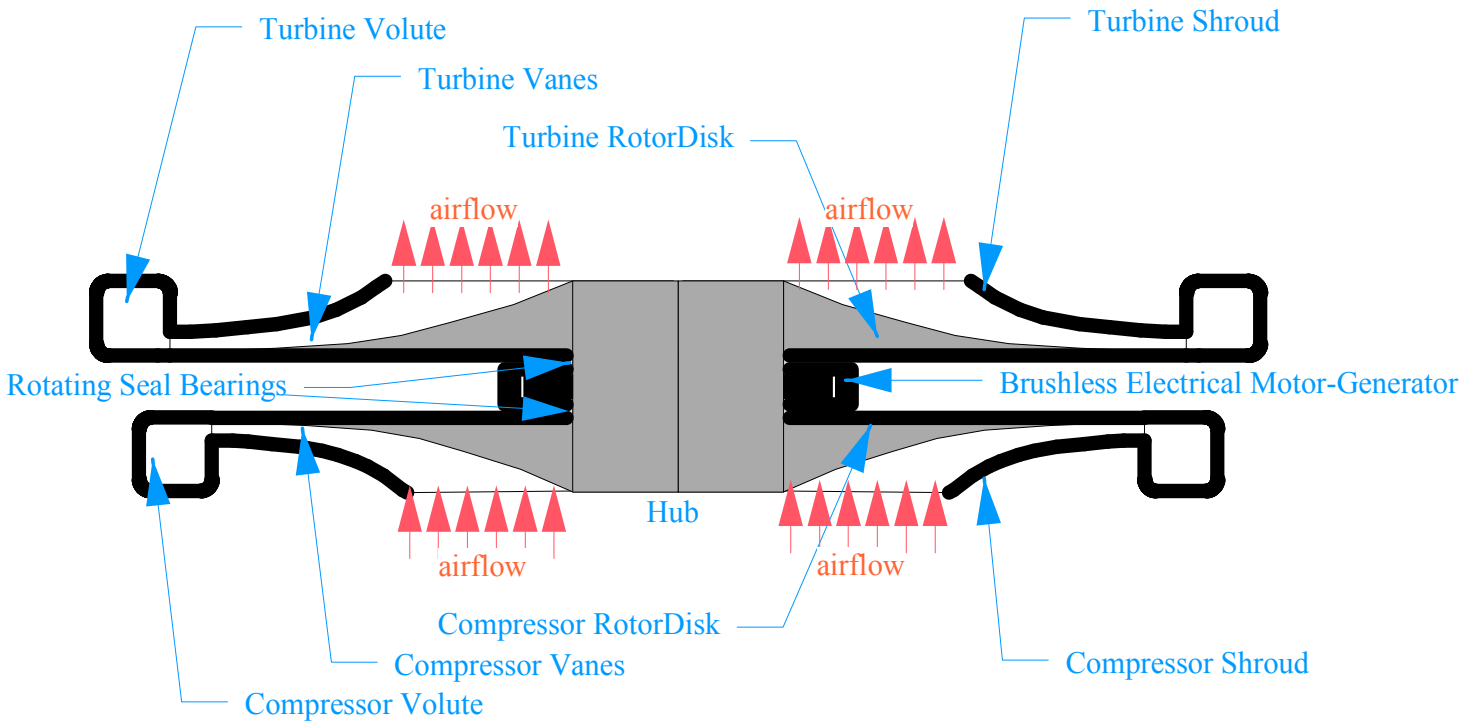


Figure 14-21: Low Pressure Turbine, Motor/Generator, and Compressor, Mounted On Common CoRotating Shaft

CHAPTER 15
15 HEAT EXCHANGER DESIGN AND ANALYSIS

The Open Brayton Cycle designs of the previous chapter would provide, as input to each heat exchanger, a flow of martian air compressed to pressures near 116 kPa and temperatures near 690 Kelvins. Each heat exchanger must then add sufficient heat to the compressed air flow to raise its temperature to about 1290 Kelvins. Also, the gas pressure exiting the heat exchanger should be 90% of the pressure at the entrance. Thus, the high pressure turbines are to be driven by 1290 Kelvins gas near 104 kPa pressure.

Heat is carried to each engine design's heat exchanger by circulating through it the liquid lithium which cools the nuclear reactor. As explained in Chapter 4, the peak local temperature of the liquid lithium in the reactor will by design not be permitted to steadily exceed 1643 Kelvins (i.e.,1370 C), since the solid refractory materials contacting the lithium have been shown to be chemically incompatible with it for long term exposure at somewhat higher temperatures, but are completely compatible up to this temperature limit. Thus, at full design point operation, the temperature of the molten lithium exiting the reactor and entering the heat exchanger should be set somewhat lower than 1643 Kelvins, by a margin designed to accommodate local temperature peaking in the reactor. Since this required margin has not yet been analyzed, it is hereby assumed that the temperature of the liquid lithium exiting the reactor is about 1500 Kelvins.

The heat exchanger could be the most compact possible if a very large lithium flow rate were provided, since then lithium would remain near 1500 Kelvins at all locations within the heat exchanger, thus also maximizing the temperatures of surfaces contacting the gas. However, such a large coolant flow rate would impact the design of reactor shielding penetrations and would also require a more massive reactor coolant pumping scheme.

An alternative heat exchanger design which has been much studied theoretically is the matched-capacity-rate counterflow design. In such a design, the two fluid flow rates are balanced so that the temperature rise of one fluid is matched by the temperature drop of the other fluid. If applied here, the molten lithium flow rate would be selected so that as the compressed air is heated to 1290 Kelvins from 690 Kelvins, an increase of 600 degrees, the lithium would be cooled from 1500 Kelvins to 900 Kelvins, a decrease of 600 degrees. Comparing the specific heats of MarsMix gas and molten lithium, it follows that each 3.45 kg/sec of MarsMix airflow would be matched to 1 kg/sec of molten lithium flow in a matched capacity rate design. Using the calculated engine design airflows of the previous chapter, the lithium flow rate would range from 0.122 kg/sec for the 100 horsepower engine to 8.25 kg/sec for the 10,000 horsepower engine.

Scoping calculations for a matched capacity rate counterflow heat exchanger were carried out using MATLAB code (see Appendix F) to evaluate and solve the differential equations. Although an entire counterflow heat exchanger is typically comprised of many parallel channels, these calculations only modeled a single channel which was

represented as indicated in Figure 15-1. Here, the cross-sectional area of the gas flow, Agas, and the "wetted perimeter" within that cross-section of the heated surface, P, are by definition related to the hydraulic diameter of the gas duct, D, as follows:

$$D \equiv \frac{4A_{gas}}{P} \qquad (15\text{-}1)$$

Gas interactions were modeled using the differential equations for high speed gas flows with both friction and heat interactions with duct walls in constant-area ducts, as developed in Saad's text, Compressible Fluid Flow, [Saad 1985, 276-288]with the adiabatic recovery factor of the duct wall's boundary layer modeled as being 100% as assumed in Saad's example calculations. Saad's formulation uses the Reynolds Analogy which links the heat transfer coefficient to the same friction factor that also causes wall friction and the resulting pressure drop. Since Saad's equations are formulated using a gas model requiring constant specific heats, it was not possible to retain the variable specific heat analysis approach which has been used herein wherever feasible. Instead, for these calculations the specific heat at constant pressure of MarsMix gas was approximated as being the divided difference of MarsMix enthalpies at 700 Kelvins and 1290 Kelvins.

Liquid lithium heat transfer was modeled using the Lyon-Martinelli correlation, and conductive heat transfer between the solid metal tubes was also modeled. However, it should not be surprising that most of the temperature drop predicted by this model occurred between the gas and its duct wall, which was always at almost the same temperature as the modeled nearby liquid lithium.

With this matched-capacity rate model, it was found that in order for the outlet pressure to be 90% of the inlet pressure after heating the air to 1290 Kelvins, the Mach number of the inlet airflow needed to be exactly 0.125, which represents a speed of 49.8 m/s at the 700 Kelvins gas inlet temperature. This Mach number result was robustly independent of the gas duct's modeled hydraulic diameter.

However, the total duct volume required to implement the heat exchanger does depend sensitively on the hydraulic diameter of the gas duct in each channel. Reducing the hydraulic diameter of each channel reduces the total required duct volume per unit air mass flow as long as its boundary layer remains turbulent. The smallest hydraulic diameter still resulting in fully turbulent flow is thus the optimum. This smallest hydraulic diameter for fully turbulent flow was found to be 0.005 meters, which provided a Reynolds number of 7087 at the duct entrance and 4621 at the duct exit, both above the 4000 level usually taken as demarking the upper end of the critical zone between laminar and turbulent flows. For this hydraulic diameter, the gas duct length needed for heating the gas up to 1290 Kelvins (with a 90% pressure ratio) was found to be 4.18 meters. If the gas duct channel's cross-sectional area matched that of a circular tube with 0.005 m diameter, then the appropriate lithium flow matching the gas capacity rate would be carried by a tube with inner diameter D_{Li}=0.002 m into which the hot lithium flows at a bulk speed of v_{Li}=0.1655 m/s.

The total duct volume needed per kg/sec of airflow was found to be 0.097 m^3. Thus, for the different engine designs summarized at the end of the previous chapter, the total required air duct heat exchanger volume ranges from 0.04 m^3 for the 100 horsepower engine to 2.76 m^3 for the 10,000 horsepower engine. Total heat exchanger volume would be slightly more than these values because of the additional volumes of the entrained liquid lithium and of the various solid metal walls. In addition, there would need to be structural provisions to support the matrix of small tubes against the forces resulting from gravity, inertia, vibration, and gas flow.

Any actual implementation of the liquid lithium-to-gas heat exchanger should use extended surfaces such as fins in order to compensate for the large ratio between the high heat transfer coefficient of a liquid and the low heat transfer coefficient of a gas. Use of extended surfaces is standard practice in liquid-to-gas heat exchangers. They provide a large external surface area contacting the gas and a small surface area contacting the liquid. Their only drawback is that they rely on conduction through the fins to internally transfer heat, so some portions of the extended surfaces will present surface temperatures with reduced differences from the gas temperature. Extended surfaces, when used to implement this type of heat exchanger, could even form the walls of the ducts carrying the compressed air. The resulting ducts do not need to be of circular cross-section. Because of the definition of hydraulic diameter as four times the area of the duct's cross-section divided by the perimeter of the duct's cross-section, other configurations are also feasible. For instance, a rectangular duct cross-section with dimensions 3 mm by 15 mm would have the same hydraulic diameter as a circular duct 5 mm in diameter, even though it has more than twice the cross-sectional area. It is also not necessary that ducts be entirely straight. Small, gentle bends in duct direction would likely have no effect on the heat exchange function. More severe bends may influence performance but the resulting performance may still be acceptable. One possibly attractive design concept would be to first wrap standard annular helical fins on circular pipes carrying the molten lithium, then enclose those fins in a pressure-tight external boundary. The result would be a rectangular cross-section duct bent into a helical shape, in good thermal contact with the liquid lithium.

A 100 mm by 100 mm gas duct cross-section including many finned tubes of inner diameter 2 mm, outer diameter 2.5 mm, and fin length 1.3 mm, appears in Figure 15-2. The choice of 7.3 mm distance between its tube centers results in the duct having an overall hydraulic diameter of 5 mm.

Various quantities calculated for the 5 mm hydraulic diameter gas duct case are shown in Figures 15-3 through 15-6. It should be noted that τ, the wall thickness parameter of Figure 15-1, was artificially increased to 10 mm in these calculations in order to conservatively compensate for not properly modeling the 2D conduction temperature drop that actually occurs in fin cross sections. Along the duct of this matched capacity rate counterflow heat exchanger, the hot wall is everywhere approximately 200 degrees hotter than the enclosed gas.

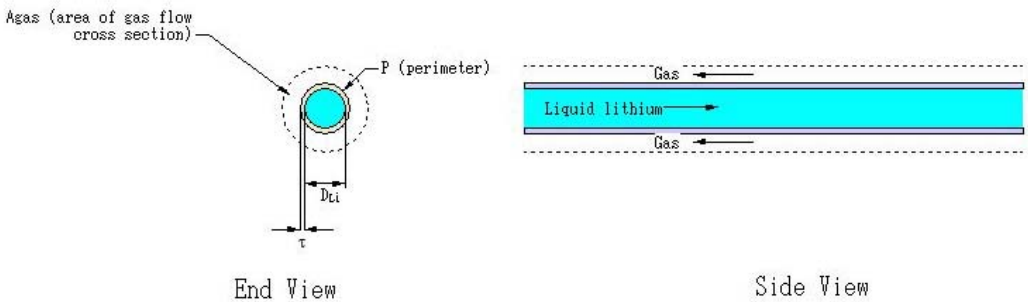


Figure 15-1: Counterflow Heat Exchanger Single Channel Model

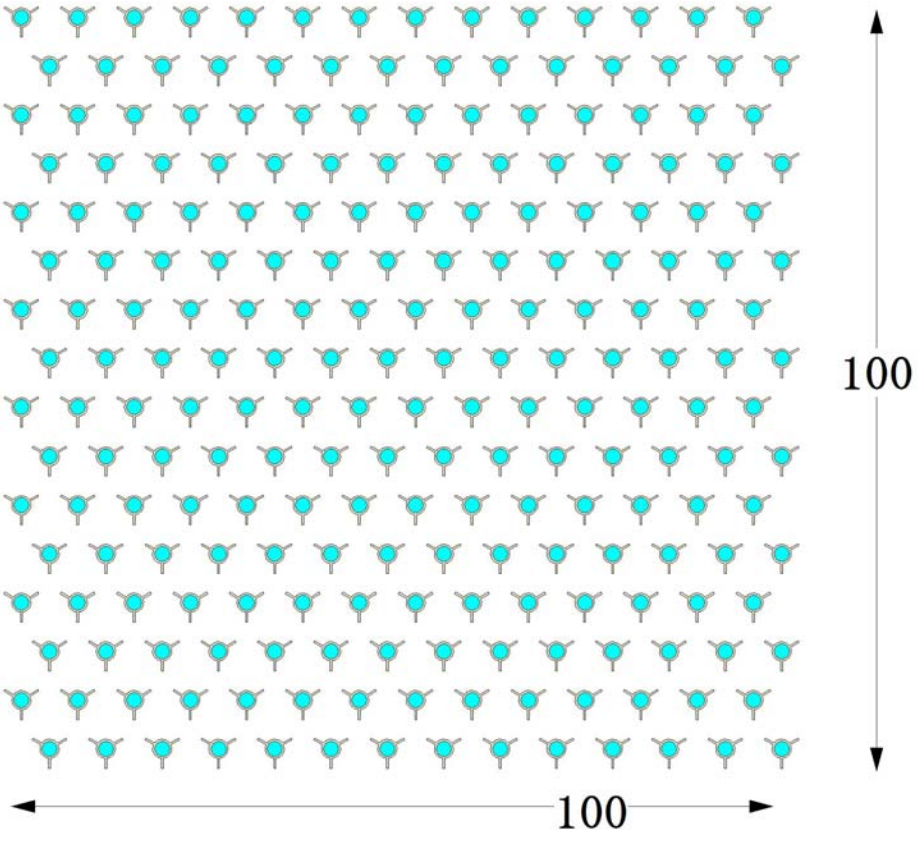


Figure 15-2: Full-Scale Multi-Channel Counterflow Heat Exchanger Cross Section

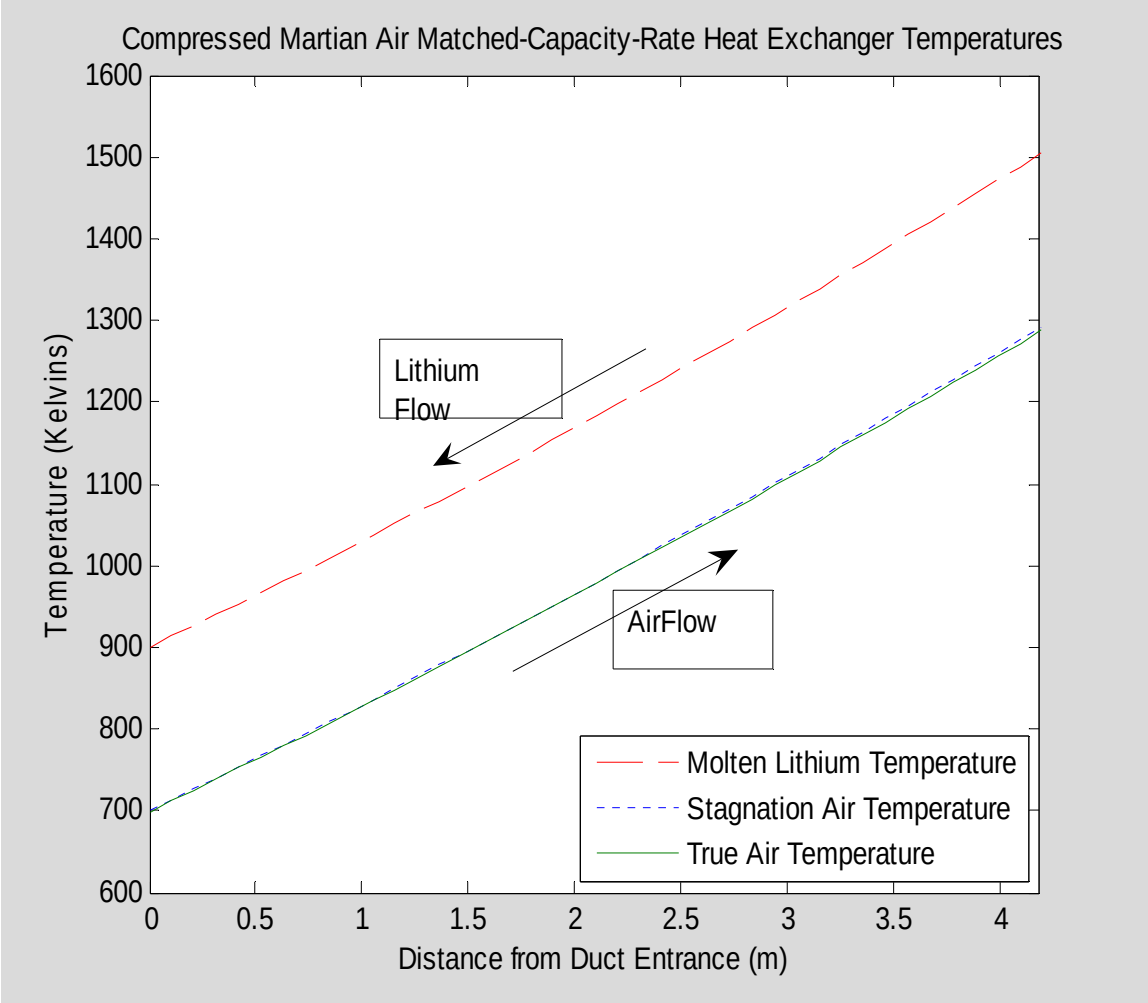


Figure 15-3: Temperatures in Optimized Matched Capacity Rate Heat Exchanger Duct

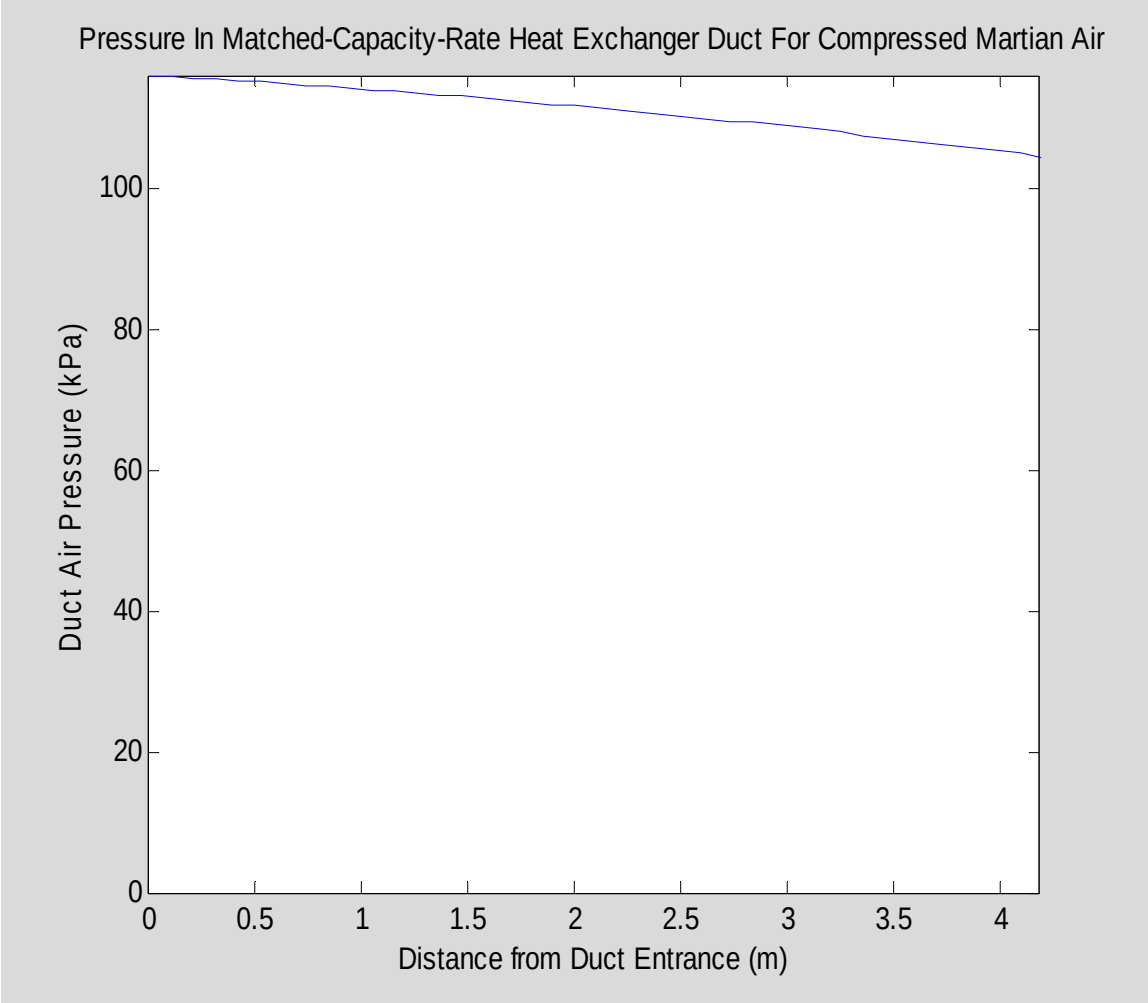


Figure 15-4: Gas Pressure in Optimized Matched Capacity Rate Heat Exchanger Duct

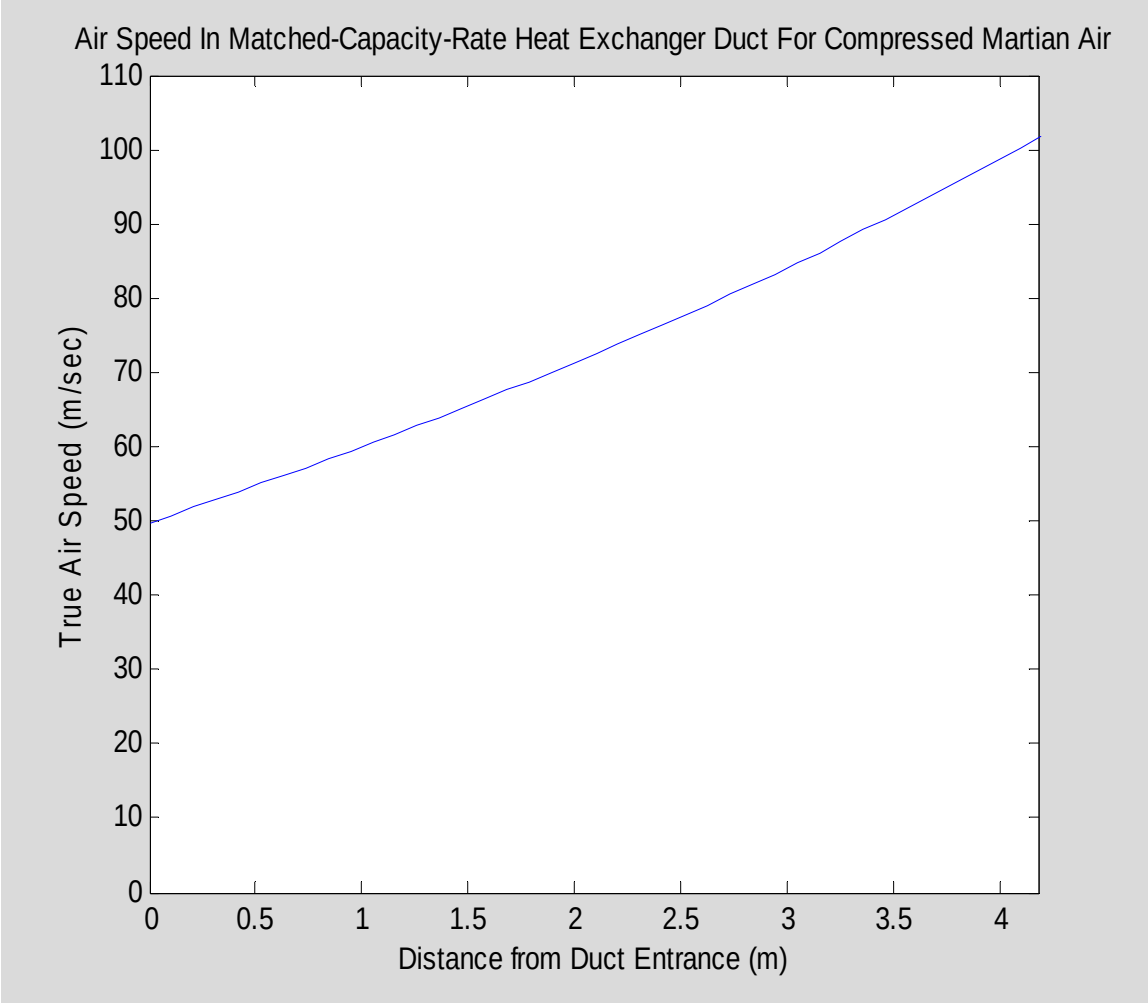


Figure 15-5: Gas Speed In Optimized Matched Capacity Rate Heat Exchanger Duct

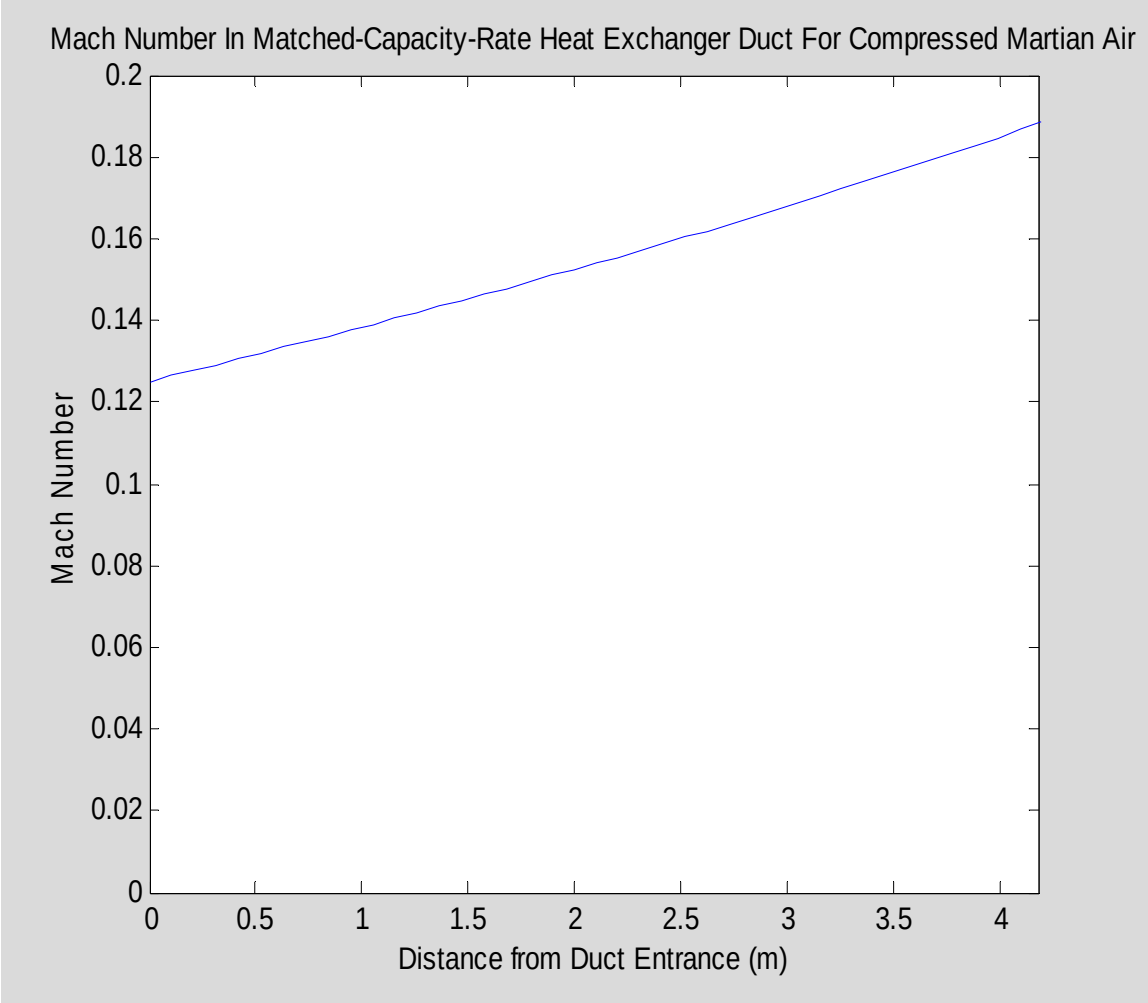


Figure 15-6: Gas Flow Speed Fraction Of Sound Speed In Optimized Matched Capacity Rate Heat Exchanger Duct

CHAPTER 16
16 CONCLUSIONS AND RECOMMENDATIONS

The present thesis effort has shown feasibility on a conceptual design level of mobile nuclear reactor engines deployed on the surface of the planet Mars, developing shaft power levels between 100 and 10000 horsepower by using their reactor heat sources to operate Open Brayton Cycle thermal conversion systems There does not appear to be any reasonable alternative way to power expected future human activities on Mars, which include water acquisition from ice mining for rocket fuel production, drinking, and industrial uses. The engines would also support general excavation and drilling in the ground, mining of other minerals, and long-distance surface travel.

Since humans will likely not visit Mars personally during the next 15 years, there should be sufficient time to carefully develop detailed nuclear engine designs for Mars in an unhurried fashion. At present, efforts to make progress on such engine designs could provide engaging training for engineering students. When human travel to Mars becomes more imminent it would be appropriate to construct and test prototype models of such nuclear engines on Earth, perhaps with US government funding through NASA and the DOE, or perhaps with funding through the governments of other space-faring nations.

LIST OF REFERENCES

REFERENCES

Balje, O. E. 1981. *Turbomachines: A guide to design, selection and theory*. New York: John Wiley & Sons. 299, 301, 307, 317, 324

Barlow, N. G. 1999. Subsurface volatile reservoirs: Clues from martian impact crater morphologies. *The fifth international conference on Mars in Pasedena, CA, July 18-23, 1999.* Lunar and Planetary Institute. 14

Bate, Roger R., Donald D. Mueller and Jerry E. White. 1971. *Fundamentals of astrodynamics*. Toronto: Dover Publications. 419, 424, 438

Bell, George I. and Samuel Glasstone. 1970. *Nuclear reactor theory*. New York: Van Nostrand Reinhold Company. 84

Bond, Victor R. and Mark C. Allman. 1996. *Modern astrodynamics*. Princeton: Princeton University Press. 429, 438, 439, 499

Borgstedt, Hans Ulrich and Cezary Guminski. 2000. Solubilities and solution chemistry in liquid alkali metals. *Monatshefte fur Chemie*. 131, 917-930. 41

Braun, R. D. and R. M. Manning. 2006 Mars entry, descent and landing challenges.. *IEEE Aerospace Conference. Big Sky Montana. March 2006.* 404

Britt, R. R. 2003. *Mars Odyssey shows intense, but managable radiation risk for astronauts*. http://www.space.com March 13, 2003. 24

Chen, S. H .P., P. C. Jain, and S. C. Saxena. 1975. Thermal conductivity and effective diffusion coefficient for vibrational energy: carbon dioxide (350-2000 K). *J. Phys. B: Atom. Molec. Phys*. Vol. 8, No. 11: 1962-72. 556

Csanady, G. T. 1964. *Turbomachines*. New York: McGraw-Hill Book Company.... 295

Decher, Reiner. 1994. *Energy conversion: Systems, flow physics and engineering*. New York: Oxford University Press. 244, 257, 263, 290

Drake, Bret G., ed. 1998. *Reference mission version 3.0 Addendum to the human exploration of Mars: The reference mission of the NASA Mars exploration study team.* Houston, TX: NASA, EX13-98-036. 404

Duderstadt, James J. and Louis J. Hamilton. 1976. *Nuclear reactor analysis*. New York: John Wiley & Sons. 48, 64, 65, 87, 796

Duffett-Smith, Peter. 1988. *Practical astronomy with your calculator*. Cambridge: Cambridge University Press. 419, 472

Duxbury, T. C., R .L .Kirk, B. A. Archinal and C. A. Neumann. 2001. Mars geodesy/cartography working group recommendations on Mars cartographic constants and coordinate systems. Abstract from '*ISPRS WG IV/9: Extraterrestrial Mapping Workshop Planetary Mapping 2001*' virtual meeting hosted by USGS, Flagstaff Arizona. Download available at: http://astrogeology.usgs.gov/Projects/ISPRS/MEETINGS/Flagstaff2001/abstracts/isprs_etm_OCT01_duxbury_A_mars_constants.pdf 472

Eckert, E. R. G. and Robert M. Drake. 1972. *Analysis of heat and mass transfer*. New York: McGraw-Hill. 556, 576, 578, 579

El-genk, Mohamed S. and Nicholas J. Morley. 1991. *Conceptual studies on the integration of a nuclear reactor system to a manned rover for Mars missions*.

Albuquerque, NM: Institute for Space Nuclear Power Studies. Final report no. UNM-ISNPS-NAG3-992. 5
Fimple, W. R. 1963. Optimum midcourse plane changes for ballistic interplanetary trajectories. *AIAA Journal* 1- no.2, (Feburary): 430-4. 395
Fraas, Arthur P. and M. Necati Ozisik. 1965. *Heat exchanger design*. New York: John Wiley & Sons. 201
Gebhart, Benjamin. 1971. *Heat transfer*. New York: Mcgraw-Hill Book Company... 65
Glasstone, Samuel 1960-3. *Quarterly status report on LAMPRE programs for period ending ... (19 reports, May1960 - Nov1963)*. Los Alamos Scientific Laboratory of the U of CA. 51
Glasstone, Samuel and Alexander Sesonske. 1967. *Nuclear reactor engineering*. New York: Van Nostrand Reinhold Company. 43, 51
Grossbeck, M. L. 2005. *NE572 - Nuclear Systems Design* class lecture notes 23 February. Universtiy of Tennessee, Knoxville, TN. 52
Gupta, G. P. and S. C. Saxena. 1970. Thermal conductivity of carbon dioxide in the temperature range 100 C to 1075 C. *Molecular physics* Vol 19, No. 6: 871-880. 556
Hanks, G. S. and J. N. Taub. 1960. *Fabrication of tantalum capsules for LAMPRE I reactor*. Los Alamos Scientific Laboratory of the U of CA. *LAMS2434*. 51
Harper, J. R. and R. Garde. 1981. *Decommissioning the Los Alamos molten plutonium reactor experiment (LAMPRE I)*. Los Alamos Scientific Laboratory of the U of CA. *LA-9052-MS*. 51
Head J. W., M. Kreslavsky, H. Hiesinger, B. Thomson and S. Pratt.. 1999. An analysis of evidence from MOLA for northern seas and oceans in the past history of Mars. *The fifth international conference on Mars. Pasedena CA. July 18-23, 1999*. Lunar and Planetary Institute 14
Ivanov, M. A. and J. W. Head. 1999 Chryse planitia, Mars: Topographic configuration from MOLA data and tests for hypothesized lakes and shorelines. *The fifth international conference on Mars. Pasedena CA. July 18-23, 1999.* Lunar and Planetary Institute 14
Jeppson, D.W., J. L. Ballif, W.W.Yuan and B. E. Chou. 1978. *Lithium literature review: lithium's properties and interactions*. Hanford Engineering Development Laboratory. HEDL-TME-78-15; UC-20 April. 45
Kiehn, R. M. 1957. *LAMPRE: a molten plutonium fueled reactor concept*. Los Alamos Scientific Laboratory of the U of CA. *LA2112*. 51
Lamarsh, John R. and Anthony J. Baratta. 2001. *Introduction to nuclear engineering*. Upper Saddle River, New Jersey: Prentice Hall. 48, 84
Lemmon, E. W. and R. T. Jacobsen. 2004. Viscosity and thermal conductivity equations for nitrogen, oxygen, argon, and air. *International Journal of Thermophysics* Vol. 25, No. 1: 21-69. 536
Lewis, E. E. and W. F. Miller, Jr. 1993. *Computational methods of neutron transport*. LaGrange Park, IL: American Nuclear Society. 139
Lide, David R. ed. 1999 . *CRC handbook of chemistry and physics*. Boca Raton, Florida: CRC Press. 556, 578
Logan, Earl Jr. 1993. *Turbomachinery: Basic theory and applications*. New York: Marcel Dekker, Inc. 307, 324

Modest, Michael F. 1993. *Radiative heat transfer*. New York: McGraw-Hill, Inc.... 218
Mueller,W. M., J .P. Blackledge and G. G. Libowitz. 1968. *Metal Hydrides*. New York: Academic Press. 812
National Bureau of Standards. 1955. *Tables of thermal properties of gases*. Washington, D.C. Circular 564..... 536
National Research Council of the National Academy of Science. 2006. *Health risks from exposure to low levels of ionizing radiation.* BEIR VII phase 2. Washington, D.C.: The National Academies Press..... 94, 97, 99
Ohse, R. W. ed. *Handbook of thermodynamic and transport properties of alkali metals*,. Oxford, Great Britain: Blackwell Scientific Publications/Alden Press. . 626, 629, 631, 633, 635, 637
Parker, T. J. and W.B.Banerdt. 1999. Observational tests of the Mars ocean hypothesis: selected MOC and MOLA results. *The fifth international conference on Mars. Pasedena CA. July 18-23, 1999*. Lunar and Planetary Institute 14
Polzin, Kurt A. 2007. Liquid metal pump technologies for nuclear surface power. *Proceedings of Space Nuclear Conference 2007 Boston MA June 24-28, 2007*..... 44
Pontryagin, L. S., V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko. 1962. *The mathmatical theory of optimal processes*. New York: Interscience Publishers - a division of John Wiley & Sons. 143, 155
Portree, David S. F. 2001. Humans to Mars: Fifty Years of Mission Planning 1950-2000. NASA-SP-2001-4521 5
Reynolds, William C. 1965. *Thermodynamics*. New York: McGraw-Hill Book Company. 518, 521
Roark, R. J. and W. C. Young. 1975. *Formulas for stress and strain*. New York: McGraw-Hill Book Company..... 312
Saad, Michel A. 1985. *Compressible fluid flow.* Englewood Cliffs, NJ: Prentice-Hall Inc. 273, 343, 793
SCALE: A modular code system for performing standardized computer analyses for licensing evaluations, ORNL/TM-2005/39, version 5, Vols. I-III, April 2005. Available from Radiation Safety Information Computational Center at Oak Ridge National Laboratory as CCC-725 60
Schaeffer, N. M., ed. 1973. *Reactor shielding for nuclear engineers*. Oak Ridge, TN: U.S. Atomic Energy Commission..... 53
Shepherd, D.G. 1956. *Principles of turbomachinery*. New York: The Macmillan Company. 324
Shultis, J. Kenneth and Richard E. Faw. 1996. *Radiation Shielding*. Upper Saddle River, NJ: Prentice-Hall, Inc..... 55
Smith, David E.,* Maria T. Zuber, Sean C. Solomon, Roger J. Phillips, James W. Head, James B. Garvin, W. Bruce Banerdt, Duane O. Muhleman, Gordon H. Pettengill, Gregory A. Neumann, Frank G. Lemoine, James B. Abshire, Oded Aharonson, C. David Brown, Steven A. Hauck, Anton B. Ivanov, Patrick J. McGovern, H. Jay Zwally, Thomas C. Duxbury. The global topography of Mars and implications for surface evolution. *Science.* 284, (28 May 1999): 1495-1503. ..... 14, 17
Stacey, Weston M. 2001. *Nuclear reactor physics*. New York: John Wiley & Sons, Inc.. ..... 48, 60, 87, 89, 112

Swickard, E. O. 1959. *Los Alamos molten plutonium reactor experiment (LAMPRE) hazard report*. Los Alamos Scientific Laboratory of the U of CA. *LA2327*. 51
Townsend, Lawrence W. 2005. Implications of the space radiation environment for human exploration in deep space. *Radiation Protection* D*osimetry* 115 (no.1-4): 44-50..... 95, 96
Turner, James E. 1995. *Atoms, radiation and radiation protection*. New York: John Wiley & Sons, Inc**.**..... 93, 96, 101
U.S. Department of Commerce. National Institutes of Standards and Technology-JANAF. 1998. *Thermochemical Tables, fourth edition,* by M. W. Chase, Jr. Gaithersburg, MD. 522
Vatulin, A.V., B. D. Rogozkin, N.M.Stepenova, Yu E. Federov, O. A. Ustinov, P. P. Poluektov, L. P. Suanov, and A. G. Sila-Novitsky. Mononitride uranium-plutonium fuel of fast lead-cooled reactors. *11th International Conference on Nuclear Engineering, Tokyo, Japan, April 20-24, 2003*. ICONE11-36414.. 52
Zinkle, S. J. and N. M. Ghoniem. 2000. Operating temperature windows for fusion reactor structural materials. *Fusion Engineering and Design* 51-52 (2000) 55-71. ... 45, 64
Zuber, Maria T., David E. Smith, Sean C. Solomon, James B. Abshire, Robert S. Afzal, Oded Aharonson, Kathryn Fishbaugh, Peter G. Ford, Herbert V. Frey, James B. Garvin, James W. Head, Anton B. Ivanov, Catherine L. Johnson, Duane O. Muhleman, Gregory A. Neumann, Gordon H. Pettengill, Roger J. Phillips, Xiaoli Sun, H. Jay Zwally, W. Bruce Banerdt, and Thomas C. Duxbury. Observations of the North Polar Region of Mars from the Mars orbiter laser altimeter. *Science*. 282 (11 December 1998): 2053-60. 14
Zubrin, Robert with Richard Wagner. 1996. *The case for Mars: The plan to settle the red planet and why we must.* New York: The Free Press a Division of Simon & Schuster. 392, 410

APPENDICES

APPENDIX A
A TECHNOLOGY FOR MANNED MARS MISSIONS

A.1 Simple Rocket Theory

Since travel between Earth and Mars will depend on rocket technology, it is appropriate to review the characteristics and limitations of rockets. Rockets function by ejecting a flow of propellant material at high speed. Their acceleration is governed by conservation of momentum as expressed by the following differential equation:

$$\frac{d\vec{v}(t)}{dt} = \vec{a}_{\text{gravity}} + \frac{1}{M(t)}\left(\vec{u}\frac{dM}{dt} + \vec{F}_{\text{drag}} + \vec{F}_{\text{other}}\right) \quad \text{(A-1)}$$

Here,

$\vec{v}(t)$ is the rocket's velocity,

t represents time,

$\vec{a}_{\text{gravity}}$ is the gravitational acceleration at the rocket's location,

M is the total vehicle mass, a decreasing function of time while the rocket engine operates,

$\vec{u}$ is the rocket's propellant exhaust velocity expressed relative to the moving rocket,

$\vec{F}_{\text{drag}}$ represents any atmospheric force accompanying movement through air,

$\vec{F}_{\text{other}}$ includes any other forces, such as contact forces with the ground prior to lift-off.

The rocket's thrust force in Equation (A-1) is

$$RocketThrust = \vec{u}\frac{dM}{dt} \quad \text{(A-2)}$$

while the total vehicle weight is

$$VehicleWeight = M\vec{a}_{\text{gravity}} \quad \text{(A-3)}$$

By neglecting gravity along with drag and other forces while also modeling the exhaust velocity as constant, the resulting simplified version of equation (A-1) can be integrated to yield the Fundamental Rocket Equation (as derived by Tsiolkovsky in 1895):

$$\Delta v = u\ \ln\left(\frac{M_{\text{initial}}}{M_{\text{final}}}\right) \quad \text{(A-4)}$$

or equivalently

$$\frac{M_{\text{initial}}}{M_{\text{final}}} = \exp\left(\frac{\Delta v}{u}\right) \quad \text{(A-5)}$$

where
Δv is the change in vehicle speed caused by the rocket engine, and
u is the rocket propellant's exhaust speed relative to the rocket.

The important rocket engine performance parameter appearing in Equations (A-4) and (A-5) is the propellant's exhaust speed, u. From Eq. (A-2), this propellant's exhaust speed is the ratio of the rocket thrust force to the propellant's mass flow rate. However, perhaps because of widespread popular confusion between units of force and mass, it is customary to define the ***Specific Impulse, I_{sp}***, as the ratio of the thrust force to the propellant's flow rate expressed in units of *weight* per second instead of mass per second. Under this convention the weight used is the propellant's weight on Earth's surface regardless of where the rocket is actually to be used. This results in the rocket's Specific Impulse performance parameter being quoted in units of time (i.e., seconds). By definition, this customary Specific Impulse rating of a rocket is proportional to the exhaust velocity:

$$u \equiv gI_{sp} \tag{A-6}$$

where the proportionality constant, g, is the acceleration of gravity at Earth's surface. Thus, the I_{sp} parameter does not contain any information beyond the exhaust speed.

Figure A-1 plots the inverse mass ratio from the rocket equation as a function of the speed ratio, over a range of parameters covering all rocket vehicles yet flown.

Figure A-1 shows that less than one percent of the initial rocket mass can theoretically be accelerated to speeds of 4.6 or more times the rocket exhaust velocity. The daunting mass ratios accompanying high multiples of exhaust speed accurately signal high associated costs of travel by rocket. However, rockets can be remarkably efficient. If we assume a rocket with constant exhaust speed starts from zero speed and neglect both gravity and aerodynamic forces, the kinetic energy imparted to its final mass is

$$E_{\text{payload}} = \frac{1}{2} M_{\text{final}} v^2 \tag{A-7}$$

while the kinetic energy invested in its propellant is

$$E_{\text{propellant}} = \frac{1}{2}\left(M_{\text{initial}} - M_{\text{final}}\right)u^2 \tag{A-8}$$

These allow the *rocket energy efficiency*, neglecting gravity and drag, to be defined as

$$\varepsilon_{rocket} = \frac{E_{\text{payload}}}{E_{\text{propellant}}} = \frac{\left(\frac{v}{u}\right)^2}{\exp\left(\frac{v}{u}\right) - 1} \tag{A-9}$$

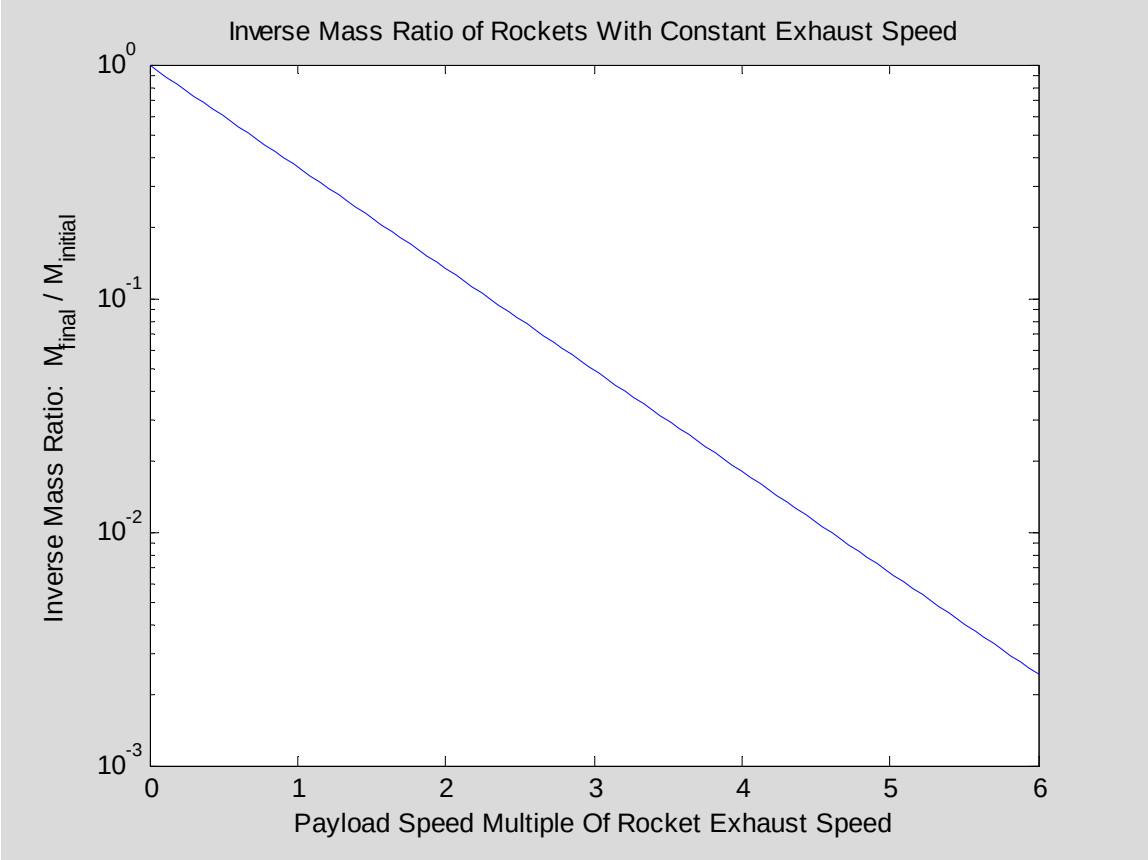


Figure A-1: Inverse Mass Ratios As Predicted by Fundamental Rocket Equation

This is plotted in Figure 5-2 for a range of speed ratios covering all rocket applications ever flown. Speed multiples between 0.23 and 4.65 have efficiencies above 20%, which shows that travel by rocket is not inherently inefficient in terms of energy consumption.

Although correct, the rocket relations discussed so far may be misleading. They should be slightly modified to separate the non-payload final mass of the rocket itself from the payload. For any single stage rocket we can identify three separate mass components as shown in the Figure A-3 diagram. These are the propulsion system's empty or "dry" mass, the propulsion system's full mass including its propellant, and the payload's mass.

The correspondences between these masses and the masses of Eq. (A-5) are as follows:

$$M_{\text{final}} \equiv M_{\text{payload}} + M_{\text{empty}} \tag{A-10}$$

$$M_{\text{initial}} \equiv M_{\text{payload}} + M_{\text{full}} \tag{A-11}$$

After substituting these definitions into Eq. (A-5) and making some simple algebraic rearrangements, the following relation results:

$$\frac{M_{\text{payload}}}{M_{\text{initial}}} = \left[\frac{\left(\frac{M_{\text{full}}}{M_{\text{empty}}} \right) \exp\left(-\frac{\Delta v}{u} \right) - 1}{\left(\frac{M_{\text{full}}}{M_{\text{empty}}} \right) - 1} \right] \tag{A-12}$$

Figure A-4 plots this relation as a function of the speed increment ratio for selected values of the propulsion system's full-to-empty mass ratio. This full-to-empty mass ratio is an important parameter characteristic of rocket systems which limits the speed increment obtainable from a single stage well below the performance suggested by Figure A-1. The difference between them occurs because rocket propulsion systems have nonzero mass fuel tanks, combustion chambers, fuel and oxidizer turbopumps and turbine drives, structural components, fairings, and for reusable systems, additional components for protection and recovery. Eq. A-12 implies the Δv obtainable from a single stage is constrained by $\Delta v \le u \ln\left(\frac{M_{\text{full}}}{M_{\text{empty}}} \right)$, with equality reached only in the impractical limit of zero payload. With a nonzero payload, the maximum Δv from a single stage rocket is even smaller. For example, with a full-to-empty mass ratio of 10, a single stage rocket can boost slightly less than half of the initial launch mass to only 60% of the rocket stage's exhaust speed.

Advocates for truly inexpensive space flight believe it will be necessary to develop fully reusable launch vehicles and then operate them with a high flight rate (e.g., daily). Today's passenger jet aircraft provide the conceptual model in which vehicles land, their fuel tanks are refilled, and then they immediately fly again. The "holy grail" sought is a

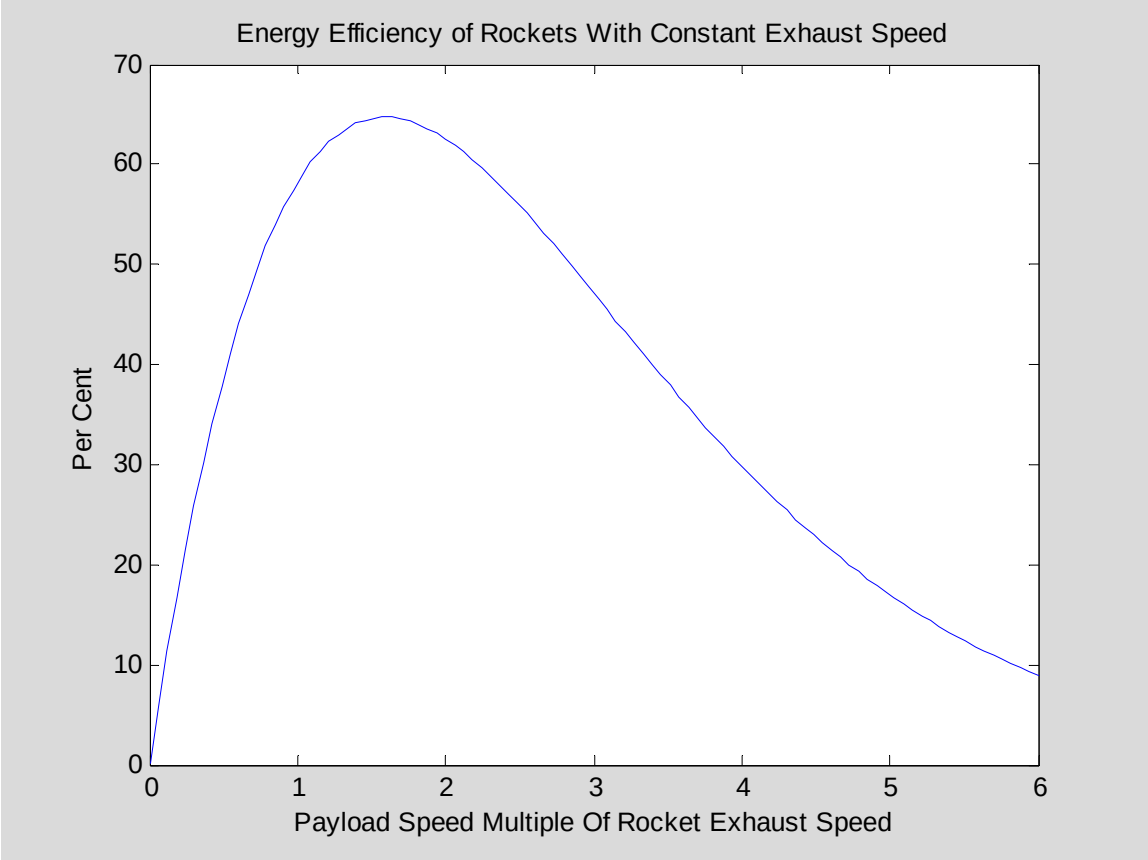


Figure A-2: Energy Efficiency of Rockets with Constant Exhaust Speed

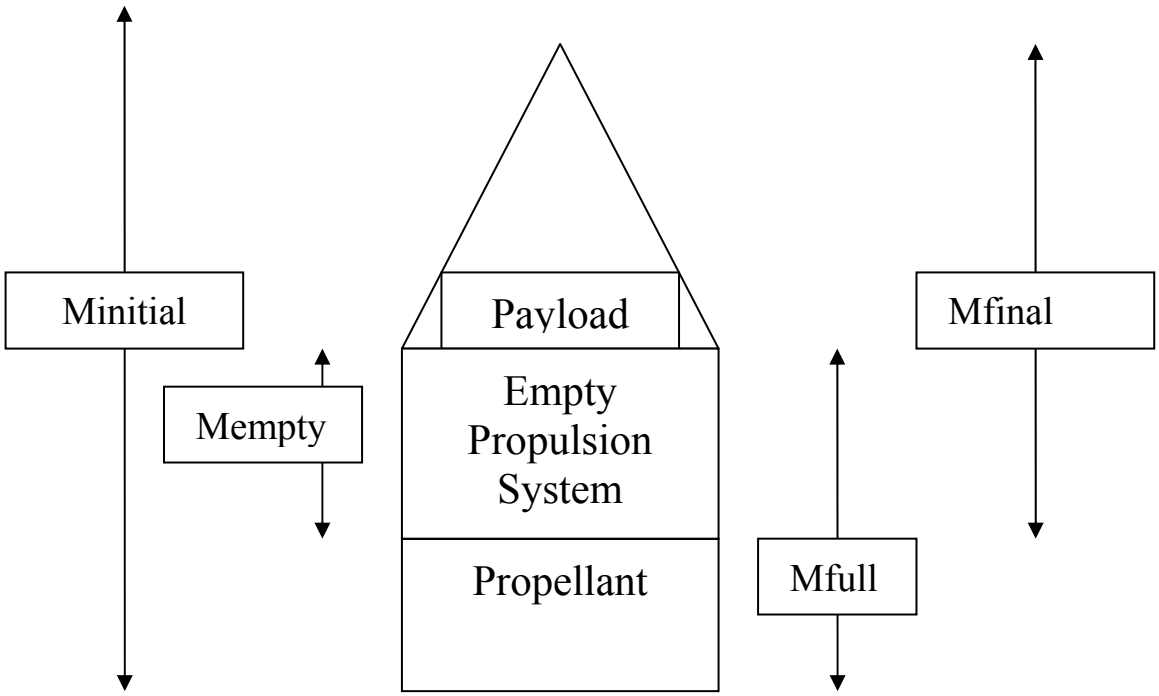


Figure A-3: Rocket Propulsion System Mass Definitions

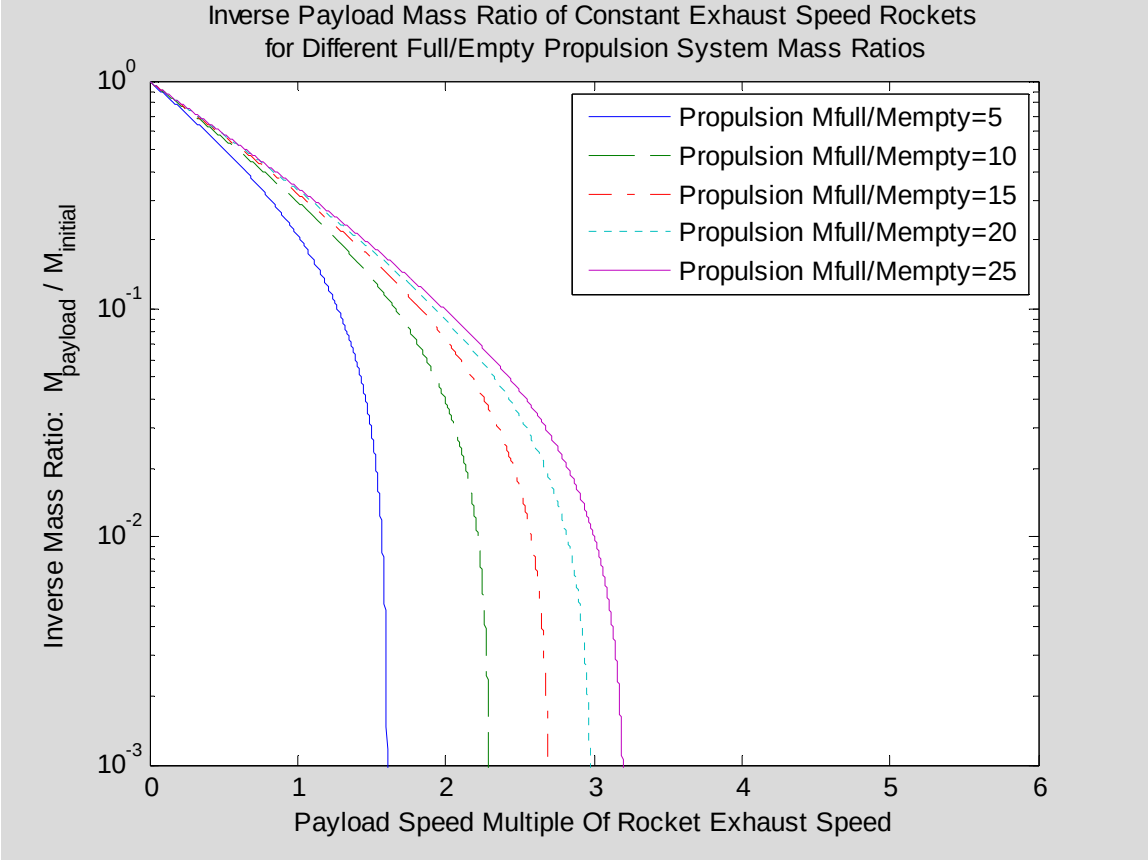


Figure A-4: Single-Stage Payload Mass Fractions

safe, fully reusable Single-Stage-To-Orbit (SSTO) rocket vehicle shuttling between Earth's surface and low Earth orbit (LEO) that additionally would not need extensive maintenance between flights. Although the overall full-to-empty mass ratio for existing reusable Space Shuttle propulsion systems is about 6, NASA studies indicated that a ratio of 10 would be needed to develop such a SSTO for Earth using conventional chemical rocket fuel. An attempt in the 1990s termed the X33 (or VentureStar) program to raise this ratio to 10 by replacing aluminum with new lighter composite materials failed when the prototype's composite material fuel tank for liquid hydrogen crumbled during temperature cycling. The X33 program was subsequently canceled.

With no SSTO existing, all space mission launches from Earth up until now have used partly expendable multi-stage rockets due to the speed increment limitations indicated by Eq. (A-12) and Figure A-4. The conventional multistage strategy adopted to overcome this speed limitation has vertically stacked multiple rocket stages and operated them in sequence without overlap, dropping each empty stage when it has finished its propellant. Most types of rocket vehicles simply discard their rocket stages after using them, making no attempt to recover them for later reuse. For such disposable stages higher values of the full-to-empty mass ratio are more typical, ranging up to about 20. A variation incorporated in the Space Shuttle design operates side-mounted booster rockets simultaneous with the main rocket engines. The Shuttle saves the main rocket engines and their associated pumps for reuse but drops the empty solid booster rockets and the empty external tank for hydrogen and oxygen fuel, with the empty solid rocket casings recovered and reused after extensive refurbishing but with the external tank discarded.
Equation (A-12) can be extended to the conventional vertically stacked multi-stage rocket systems. For such a N-stage rocket,

$$\frac{M_{\text{payload}}}{M_{\text{initial}}} = \prod_{i=1}^{i=N} \left[\frac{\left(\frac{M_{\text{full}}}{M_{\text{empty}}} \right)_i \exp\left(-\frac{(\Delta v)_i}{u_i} \right) - 1}{\left(\frac{M_{\text{full}}}{M_{\text{empty}}} \right)_i - 1} \right] \tag{A-13}$$

where

$$\Delta v_{\text{total}} = \sum_{i=1}^{i=N} (\Delta v)_i \tag{A-14}$$

and the index, i, refers to the stage number in the sequence of operation.

A.2 General Rocket Technology

Clearly, the exponential function in the rocket equation can lead to impractical mass ratios and associated high cost. To avoid them the ratio of the payload's velocity increment to the rocket's exhaust speed must be limited. Therefore, high rocket exhaust speeds are desirable and high mission velocity increments should be avoided.
Extremely high rocket exhaust speeds are possible in space if the rocket propellant is electrically accelerated, e.g., by ion-electrostatic or by plasma-magnetohydrodynamic

schemes. In principle these could propel spacecraft with vastly less propellant mass. Such schemes have been successfully tested in laboratories and some prototypes have been tested in space. All such electrical rockets require very high electrical power and develop a low but continuous thrust force. These technologies have not yet become competitive because presently available power supplies to meet their electrical needs are more massive than the conventional rocket systems they would replace. The specific power (i.e., power per unit mass) of today's solar array technology is too low to be competitive at Earth's distance from the sun and is even lower at the greater distances involved in travel to Mars. Specific powers of existing nuclear power designs developed in the past are even lower than solar arrays, largely due to the massive thermal radiators they employ for thermal conversion waste heat rejection. Although they are not useful today and will probably not be used in manned missions to Mars during the next few decades, such electrical rockets may perhaps eventually prove useful for long duration space missions after future technological progress dramatically increases the specific power available from nuclear reactors in space.

However, very high exhaust speed electrical rockets will never be used for launch. For a vertical launch from the surface of a planet the rocket's initial thrust force must exceed the total initial weight of the vehicle. All rockets yet designed or proposed for launching vehicles from surfaces of solar system bodies (e.g., Earth, Luna, or Mars) are thermally based and develop their high propellant exhaust speeds by expanding a high temperature, high pressure gas through a specially shaped (Laval) exit nozzle. Chemical reactions have provided the high temperature and pressure in all such rockets actually flown. Nuclear-thermal designs in which a gas (usually hydrogen) is heated by a nuclear fission reactor have been tested on the ground but have never flown.

Table A-1 lists exhaust speeds and the I_{sp} parameter for various rocket technologies.

The only technology promising hope of a significant improvement over present thermal rocket exhaust speeds is the nuclear-fission-powered thermal rocket using pure hydrogen as propellant. Because of the low molecular mass of hydrogen, rocket exhaust speeds attainable with hydrogen gas temperatures in the usable range of available solid heat-transfer materials exceed 10 km/sec. Project NERVA tested one such scheme on the ground in the late 1960s. More aggressive but completely untested nuclear thermal rocket design concepts include Westinghouse's "nuclear lightbulb" proposal in which a specially confined *gaseous* nuclear reactor would heat hydrogen to even higher temperatures by radiative heat transfer in order to achieve exhaust velocities approaching 20 km/sec. However, although nuclear thermal rocket schemes show promise it is not clear whether they will live up to that promise. Technical problems must be solved to optimize heat transfer and provide adequate radiation shielding. It is not clear that the resulting nuclear thermal rocket systems will be light enough to take off from a planet. In any case, the facts that today few researchers are working on these problems and that there are no significant funded efforts to develop nuclear thermal rockets imply they will not be available for early manned missions to Mars. They are therefore ignored.

Table A-1: Comparisons of Noteworthy Rocket Technologies

Rocket Technology	Exhaust Speed, u, (km/sec)	Specific Impulse, I_{sp}, (sec)	Comments
Cold Pressurized Gas	0.6	61	Limited uses, e.g., for attitude control
Liquid Monopropellant	1.4-2.35	14A-241	Limited uses
Hydrazine(N_2H_4)/Nitrogen Tetroxide (N_2O_4)	2.8-3.2	287-328	Stable at room temperature/pressure, toxic; used in Lunar Lander, Shuttle's Orbital Maneuvering System, deep space probes
Liquid Oxygen /Kerosene	3.0-3.4	307-348	Saturn 5 First Stage, China's Long March
Liquid Oxygen /Methane	3.7-3.9	379-399	Never flown; proposed for Mars missions
Liquid Oxygen /Hydrogen	4.0-4.6	410-471	Upper stage Saturn 5, Atlas Centaur, Shuttle, Europe's Ariane, Russia's Energya
Liquid Fluorine/ Lithium/Hydrogen	5.32	542	Corrosive and toxic; tested but never flown.
Solid Rocket	2.6-3.1	266-317	Missiles, Shuttle Boosters
Liquid Oxygen/Solid Fuel Hybrid Rocket	2.9-3.5	297-358	Never flown.
Nuclear Fission Thermal/ Hydrogen	8-20	820-2050	Project NERVA rocket tested but never flown

More capable chemical rocket fuels will certainly not appear in the future since all chemical rocket fuels with the highest possible exhaust speeds have already been investigated. The highest energy chemical fuels, using liquid fluorine as an oxidizer, are excessively corrosive to rocket engines and thus impractical in addition to being toxic and environmentally onerous. The best of the practical combinations burns liquid hydrogen (LH_2) in liquid oxygen (LOX) to exhaust water vapor while other popular propellant combinations burn hydrocarbons in LOX to exhaust water vapor and carbon dioxide. A difficulty blocking past applications of these far from Earth has been that they use cryogens. Current practice requires cryogens such as LH_2 or LOX to be loaded into a rocket's fuel tanks immediately prior to launch from Earth since rocket fuel tank thermal insulation is typically kept thin to conserve mass. Better low-mass thermal insulation and compact flight-qualified active cryogenic refrigeration systems will need to be developed in order to use cryogenic fuels far from Earth.

Technical problems with cryogen storage have been avoided by using chemical fuels stable at room temperature and pressure. There are two approaches, one using solid fuel and the other using liquid bipropellants. The drawback both is their comparatively low exhaust speeds, limiting them to applications needing only small speed increments.

Solid rockets, storable for years without special provisions, are used in military missiles, in the Shuttle's side-mounted booster rockets during the first 50 km of low-speed climbing, and in the motors used to circularize geosynchronous orbits of communication satellites. According to NASA the Shuttle's solid rocket fuel is 69.93% ammonium chlorate oxidizer, 16% powdered aluminum fuel, and 12.04% polybutadienne acrylic acid binder, with epoxy curing agent and catalyst making up the other 2%. These form a uniform mixture with the consistency of a rubber pencil eraser filling the steel cylindrical rocket casing except for special slots designed to optimize burn rates. Solid rockets cannot be turned off after they are ignited; they burn until their fuel is gone.

The most popular bipropellant combination uses nitrogen tetroxide as the oxidizer and monomethyl hydrazine as fuel (called MMH/NTO in NASA documents). Both are stable liquids at room temperature and pressure, storable for years without special provisions. In addition, they have the convenient *hypergolic* property, i.e., they ignite on contact with each other without requiring any special ignition system. Rockets using these can be turned on and off by simply controlling liquid flow. They were used as the sole fuel for lunar landers in Apollo missions and have been used in most deep space probes to other planets. They are also used as the sole fuel for the Shuttle's Orbital Maneuvering System (OMS) which takes over from the Shuttle's LH_2/LOX main engines after reaching orbit.

To put the performance of available rocket technologies in context it is necessary to consider the mission requirements for their application. To this end, various characteristic speeds of selected solar system bodies of interest are listed in Table A-2.

For launches into Earth orbit, the aerodynamic and gravity launch loss terms appearing in Equation (A-1) but neglected in Eqs (A-4) and (A-5) are minimized by restricting

maximum speed during passage through denser parts of the atmosphere while also accelerating at several times Earth's gravitational acceleration. Rockets also must overcome the potential energy needed to climb above the atmosphere to where orbits are relatively stable (i.e.., about 200 km). With optimized tradeoffs of those energy effects, the total Δv to be supplied by rocket to reach a 200 km low Earth orbit increases to about 9 km/sec, depending on the launch rockets used. Another 3.22 km/sec is needed to reach the 11 km/sec needed at 200 km altitude to escape Earth's gravity entirely.

In Apollo manned missions to Luna ending 35 years ago, a Saturn 5 rocket provided the 11.2 km/sec plus launch losses needed at Earth's surface to enter a translunar trajectory, then the Apollo command module's rocket engine reduced speed by 0.7 km/sec to enter low lunar orbit. The lunar lander's rockets provided the 1.7 km/sec needed for a soft landing on the lunar surface, then provided another 1.7 km/sec in order to rejoin the command module in lunar orbit. Finally, the command module's rocket engine increased speed by 0.7 km/sec to enter a trajectory returning to Earth where aerodynamic friction instead of rockets slowed the vehicle from its 11.2 km/sec reentry speed. Thus, the total Δv supplied by rocket for each Apollo mission was about 11.2+ 0.7+1.7+1.7+0.7=16.0 km/sec plus launch losses for the two astronauts visiting the lunar surface, and 11.2+0.7+0.7=12.6 km/sec plus launch losses for the third astronaut who stayed with the command module in low lunar orbit. In light of rocket theory, a comparison of rocket exhaust speeds listed in Table A-1 with Apollo's weighted average of 12.6 km/sec and 16.0 km/sec total rocket Δv shows a large mass ratio was necessary.

A.3 Past Rockets Relevant for Manned Missions to Luna and Mars

Many rockets large enough to launch people into low Earth orbit (LEO) have flown. However, to launch either manned missions or relevant massive cargoes to accompany manned missions towards either Luna or Mars, a rocket system must be able to place about 100 tonnes or more in LEO. Only three such heavy-lift rocket systems have ever been developed, i.e., the Saturn 5, Shuttle, and Energiya.

A.3.1 Saturn 5

After the 25 May 1961 speech by President John Kennedy promising the US would land Americans on the moon during that decade, Werner Von Braun and his team of former Nazi rocket engineers from Peenemünde were tasked, along with the new NASA organization, with developing a moon rocket. The resulting Figure A-6 Saturn 5 system developed for Apollo was a three-stage rocket system 281 feet tall and 33 feet in diameter at its first stage base, producing a sea-level launch thrust of 7,653,854 pounds (i.e., $3.4094 * 10^7$ Newtons). Fully fueled and with its Figure A-5 Apollo module payloads mounted on its top its launch mass was 6,457,250 lbm (i.e., 2,942,939 kg). The Saturn 5 was capable of launching a payload mass of 125 English tons (i.e., 113.6 tonnes) into a circular low earth orbit of altitude 105 nautical miles (i.e., 194.4 km), alternatively able to launch 50 English tons (45.5 tonnes) into lunar transfer orbit. Table A-3 lists parameters of the Saturn 5 and Apollo module components.

Table A-2: Characteristic Orbital Speeds for Selected Bodies in Solar System

Speed Item	km/sec
Circular orbit around Earth at "treetop level", relative to Earth	7.91
Circular orbit around Earth at 200 km altitude, relative to Earth	7.78
Escape speed from Earth surface, relative to Earth	11.19
Escape speed from 200 km altitude, relative to Earth	11.00
Earth's mean orbital speed around Sun, relative to Sun	29.78
Circular orbit around Luna (Earth's moon) near its surface, relative to Luna	1.68
Escape speed from lunar surface, relative to Luna	2.38
Luna's mean orbital speed around Earth, relative to Earth	1.02
Circular orbit around Mars at "treetop level", relative to Mars	3.57
Circular orbit around Mars at 200 km altitude, relative to Mars	3.47
Escape speed from Mars' surface, relative to Mars	5.03
Mars' mean orbital speed around Sun, relative to Sun	24.13

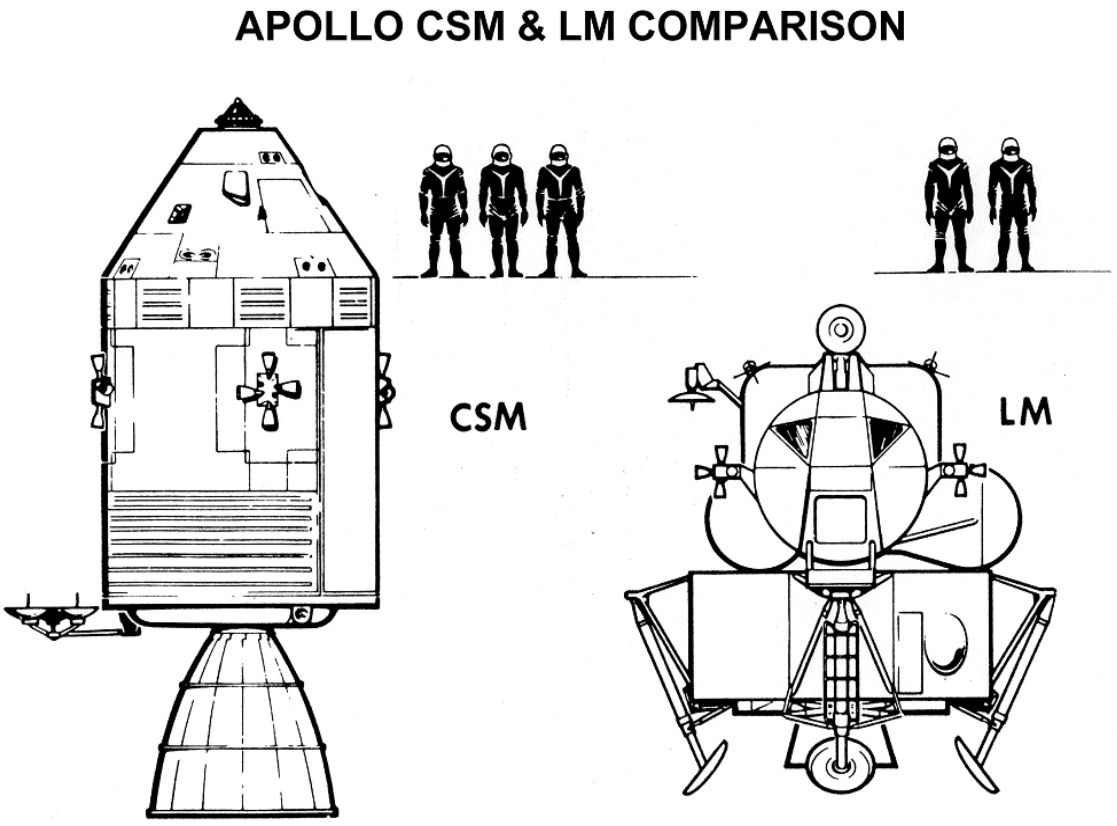


Figure A-5: Apollo Modules (Courtesy NASA)

Figure A-6: Apollo 11 Saturn 5 Launch for First Lunar Landing (Courtesy NASA)

Table A-3: Masses of Saturn 5 Rocket And The Apollo Modules It Launched

Item	**Propellant**	**Empty Mass (lbm)**	**Full Mass (lbm)**	**Full/Empty Mass Ratio**
Command Module	N/A	N/A	12,250	N/A
Service Module	N_2H_2/N_2O_4		51,243	
Lunar Module Ascent Stage **& Cabin**	N_2H_2/N_2O_4	4,804	10,622	2.211
Lunar Module Descent Stage	N_2H_2/N_2O_4	4,483	22,583	5.037
Saturn 5 3rd Stage & Interstage	LH_2/LOX	33,081	260,523	7.875
Saturn 5 2nd Stage & Interstage	LH_2/LOX	90,021	1,059,171	11.766
Saturn 5 1st Stage	RP1 Kerosene/ LOX	288,750	5,022,674	17.395
Launch Abort Escape Tower			8,910	N/A
Total Mass			**6,457,250**	

The three linked Apollo modules that the Saturn 5 rocket injected into lunar transfer orbit (See Fig. A-5) included (1) the cylindrical Service Module (SM) containing fluid supplies and a rocket propulsion system to enter and leave low lunar orbit, (2) the conical Command Module (CM) with its heat shield of brazed stainless steel honeycomb covered by ablative phenolic epoxy resin, parachutes for an ocean landing, and with its 5.9 m^3 pressurized cabin which the A-man crew shared with equipment, and (3) the Lunar Module (LM) containing a 4.5 m^3 cabin, a descent stage to reach the lunar surface and an ascent stage to leave it. Table 3.3 lists system masses.

The first manned circumlunar mission (Apollo 8) was carried out in December 1968 and another (Apollo 10) occurred in early 1969. The first lunar landing, Apollo 11, occurred in July 1969 and was soon followed by an Apollo 12 landing. The Apollo 13 mission suffered an explosive accident which its crew barely survived. Only four more Apollo missions were conducted, Apollos 14, 15, 16, and 17. After the near-disaster of Apollo 13, then-President Nixon ordered the lunar program curtailed and ordered NASA to stop considering missions to Mars, halted production of Saturn 5 systems and even canceled the final two lunar flights (Apollo 18 and 19) for which the hardware had already been produced. He also canceled Project NERVA, an effort to develop nuclear thermal rocket systems. About 15 years later, after the USSR tested its Energiya rocket, an internal NASA investigation into restarting Saturn 5 production failed to find its blueprint drawings and the Saturn 5 design team by that time was no longer available.

A.3.2 Shuttle STS

The Shuttle system places in low Earth orbit a mass large enough to be relevant for manned missions to Luna or Mars, but its particular configuration makes it not directly usable for such missions. The Space Shuttle Space Transportation System (STS) was designed in the early 1970s, immediately after the Apollo lunar program was halted. The STS was the first design deployed to access low Earth orbit via reusable rocket systems. It was designed to carry freight in both directions between Earth's surface and Earth orbits with altitudes ranging from 185 km to 643 km. Its orbiter (with *empty* mass 173,800 lbm = 79 tonnes without rockets and a size similar to a DC-9 airplane) is dominated by its huge cargo bay, large enough to carry and protect cylindrical payloads with diameters exceeding 4 meters and axial lengths exceeding 18 meters. In its original design the mass rating of its cargo bay payload was 65,000 lbm (i.e., 29.5 tonnes) but NASA's web sites state that upgrades since the original design have added another 7.3 tonnes to its rated cargo capacity. The typical total mass put into low Earth orbit in each shuttle mission, including both cargo and orbiter together, is about 125 tonnes.

The orbiter includes a 65.8 m^3 pressurized cabin volume intended to house between 2 and 7 crew members for orbital flights of up to 17 days duration, and an airlock permitting spacewalks without depressurizing the cabin. The most novel aspect of the orbiter is that, except for the external fuel tank, it houses all other components of the main LH_2/LOX rocket engine propulsion system (including fuel pumps and turbines) so that these expensive components can be reused for many flights. The winged aluminum orbiter reenters the atmosphere protected by its non-ablative, and thus reusable, thermal

protection system of ceramic tiles, controllably glides up to 3000 km *sideways* from its orbital path in order to reach a convenient landing site, then makes a pinpoint landing like an airplane on a runway. On the runway the orbiter stands 57 feet tall, is 122 feet long and has a wingspan of 58 feet. In its launch configuration of Figure A-7 the STS stands 184 feet tall, has a lift-off mass of 4,500,000 lbm (i.e., 2045 tonnes), and develops a sea-level thrust at launch of 7.8 million pounds (i.e., 3.5 $*10^7$ Newtons). Table A-4. lists its propulsion system parameters.

Of the six orbiters constructed, one was for evaluation and never flew while two others were destroyed by accidents in which their crews were lost. Subsequent inquests blamed the accidents on mismanagement. The first accident resulted from NASA management ignoring warnings from engineers that solid booster rocket seals would not work reliably if too cold. After the first accident, NASA disallowed rocket fuel in the shuttle's cargo bay, which had the effect of making Shuttle useless for space missions to Luna or other planets. The second accident resulted from NASA's 20 year management refusal to address the safety hazard of foam falling from the external tank and damaging thermal protection system tiles during most launches. The remaining three orbiters are still in use to finish building the International Space Station (ISS).

A.3.3 Energiya

In 1976 the USSR initiated parallel developments of the reusable "Buran" space shuttle which resembled the US space shuttle and of a heavy-lift rocket launcher, "Energiya", to place Buran into orbit. The resulting initial version of the Energiya rocket was able to launch 80 tonnes into low Earth orbit, but other drawing-board versions of Energiya planned for later development were expected to be able to launch up to 175 tonnes. The original design version flew only twice, on 15 May 1987 for a test flight and again in 1988 when it launched the unmanned Buran orbiter in its first and only orbital flight (which was successful). The subsequent political break-up of the USSR altogether halted Energiya production, testing, and operation due to the cessation of government funding.

A.4 The Future Ares Rocket Systems For Travel To Luna and Mars

In the wake of the second Shuttle accident, its following inquest and the subsequent commission reviewing US space policy, President George W. Bush announced a changed policy emphasis termed the "Vision for Space Exploration". Although the International Space Station construction would still be finished and would still be followed by the previously planned ISS experimental program in concert with Japan, Canada, Europe, and Russia, the planning of manned missions first to Luna and later to Mars would be resumed and eventually carried out for science and exploration purposes. However, NASA's annual funding levels, which in "real inflated dollars" are far less than its historical funding levels during the Apollo program, would not be increased. Instead, the new space travel capabilities would be paid for over time by wise technical choices and judicious use of new technology, by retiring unneeded capabilities, and if necessary, by taking considerable time before missions to Luna and Mars would actually begin.

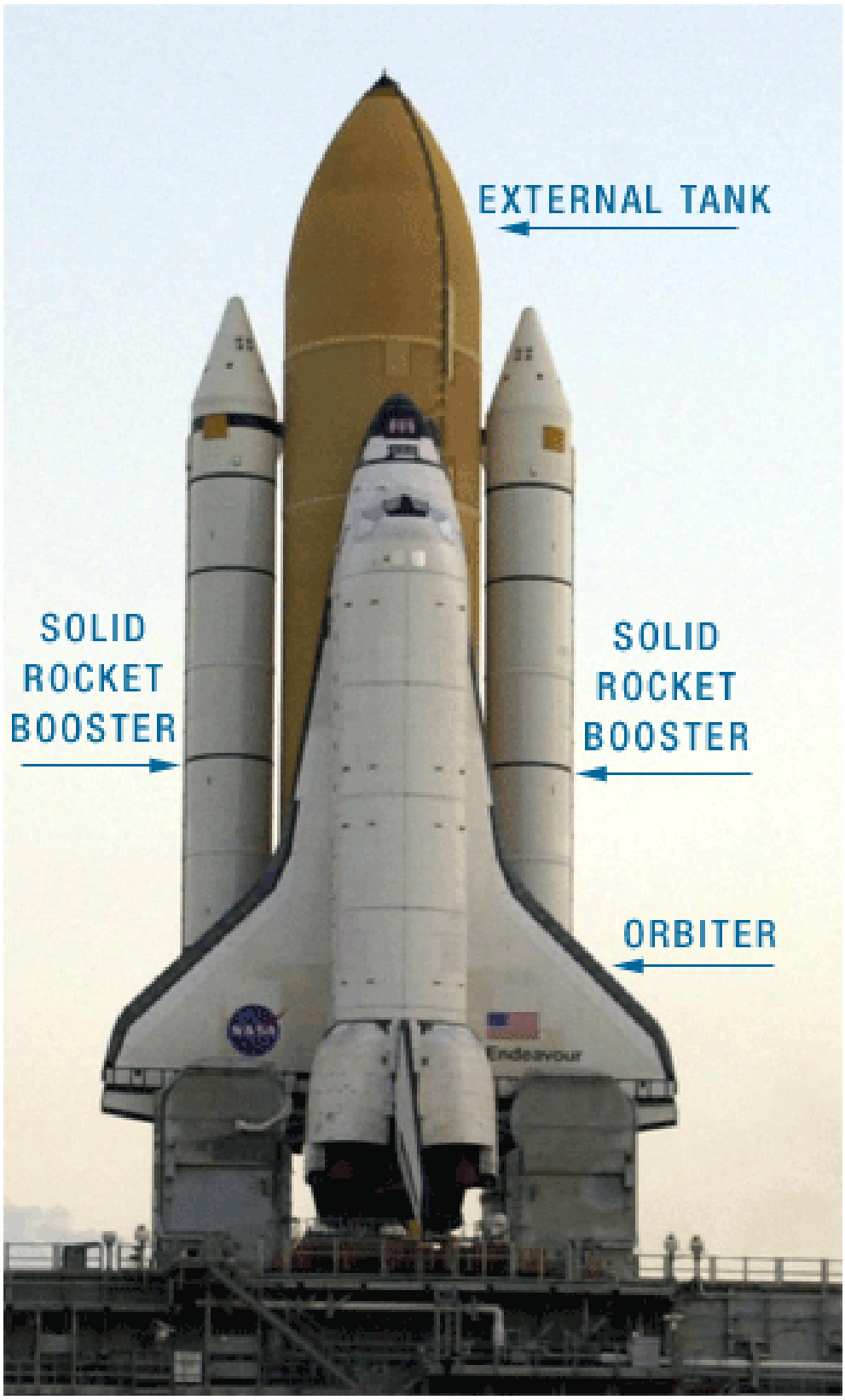


Figure A-7: Shuttle Space Transportation System(Courtesy NASA)

Table A-4: Space Shuttle Space Transportation System Propulsion Components

Item	**Propellant**	**Empty Mass (lbm)**	**Full Mass (lbm)**	**Full/Empty Mass Ratio**
Solid Boosters	$Al/NH_4ClO_3/$	384,000	2,600,000	6.771
External Tank	LH2/LOX	66,000	1,655,600	25.1
Orbiter	N_2H_2/N_2O_4	173,800	220,000+Cargo	1.266
Total Mass			**4,475,600+Cargo**	

This change is a challenge for the NASA organization, which has not designed any new manned spacecraft since 1973. Few present NASA engineering employees can recall working on design development of either the Saturn 5 or Shutttle. NASA's present budget-limited management plan in response to this policy shift is to retire the Shuttle (and its entire technical operations staff) in the year 2010 after first finishing ISS construction. A new space transportation system code-named "Constellation" will then be funded and developed for launches from the Kennedy space center in Florida to provide both crew transport between Earth and the ISS and also crew transport between Earth and a base to be constructed on Luna. To save on engineering development funds, Constellation will return to using expendable rockets, thus abandoning the quest for low-cost reusable systems. The presently articulated schedule calls for a manned lunar landing by the year 2020. NASA planners hope that much of this new system can also be used for a sequence of manned missions to Mars commencing later during the 2020s.

At the time of this writing the Constellation project has not advanced very far beyond the conceptual design phase that generates artistic drawings, some of which are included here as Figures A-8 through A-11. The Constellation design project management has so far adopted module designs resembling Apollo's conical CM and cylindrical SM but photographically enlarged by about 30%. The diameter of the conical "Orion" CM has now been set at 16.5 feet (about 5.0 meters). The resulting cabin volume of this new 25 tonne craft is more than twice that of Apollo's CM (but far less than the cabin volume of Shuttle). The designed Orion will be able to carry six people in somewhat cramped circumstances between Earth and the ISS, and is planned to also accommodate four people for round-trip travel between Earth and low lunar orbit. Although it represents an abandonment of Shuttle’s capability for precision landings on Earth, the conical shape of the Orion CM was chosen as being safer than a winged shape in the high speed atmospheric entries into Earth's atmosphere needed for returns from Luna or Mars. Perhaps more importantly, it avoids engineering costs of evaluating alternatives. (For eventual 6-person Mars missions, additional crew space would need to be provided by an expendable Mars Transfer Vehicle). Unlike Apollo, Orion has adopted extendable solar arrays similar to Russia's Soyuz vehicle and also has thermal radiator temperature control provisions similar to the ISS. It is envisioned that Orion will orbit Luna unoccupied for up to six months while its passengers visit the lunar surface using a lunar lander (LSAM). The Orion CM will have an ablative heat shield for Earth atmospheric entry similar to Apollo's CM but also will have provisions for landing on solid ground similar to Soyez modules instead of in the ocean. Self-propulsion of all Constellation modules (CM, SM, LSAM) will use N_2H_2/N_2O_4 rocket fuel. The Lunar Surface Access Module (LSAM) lunar lander will carry the four people to and from a base location on the lunar surface. The concept requires the base location to already hold suitably large pressurized modules to serve as habitats along with assorted other equipment and ample supplies, all previously delivered by unmanned one-way cargo rockets in support of a sequence of lunar missions. NASA recently announced this lunar base will be located at Luna's south pole where solar power is always available, beside the always-dark Aitken Impact Basin where the Lunar Prospector orbiter's neutron spectrometer measurements indicated extensive deposits of water-ice may exist.

Figure A-8: Orion CM & SM With ISS (Courtesy NASA)

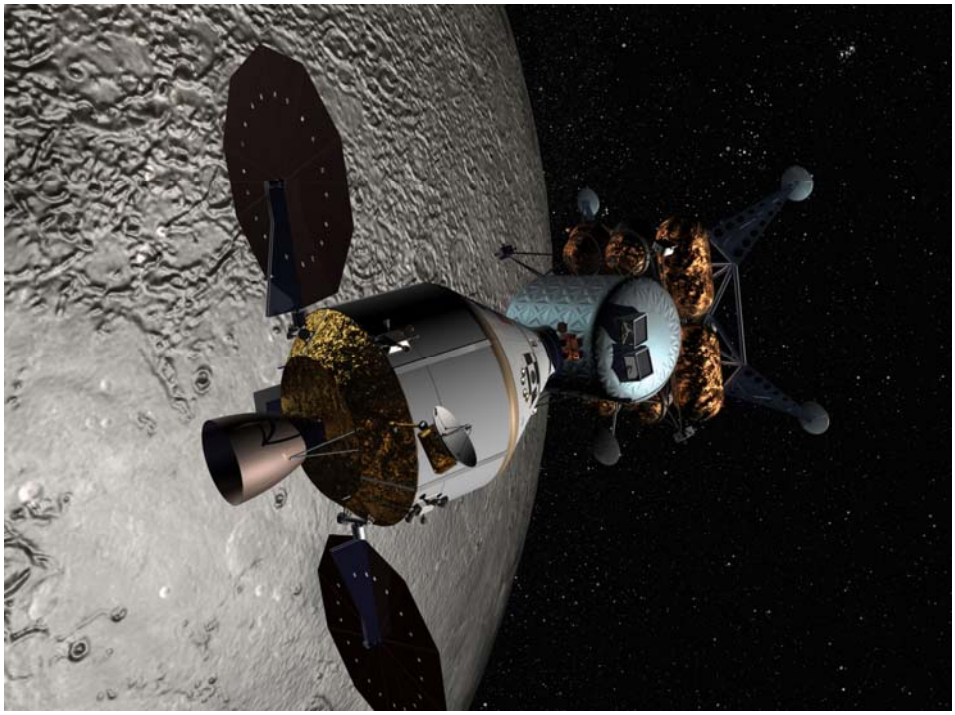


Figure A-9: Orion & LSAM At Luna (Courtesy NASA)

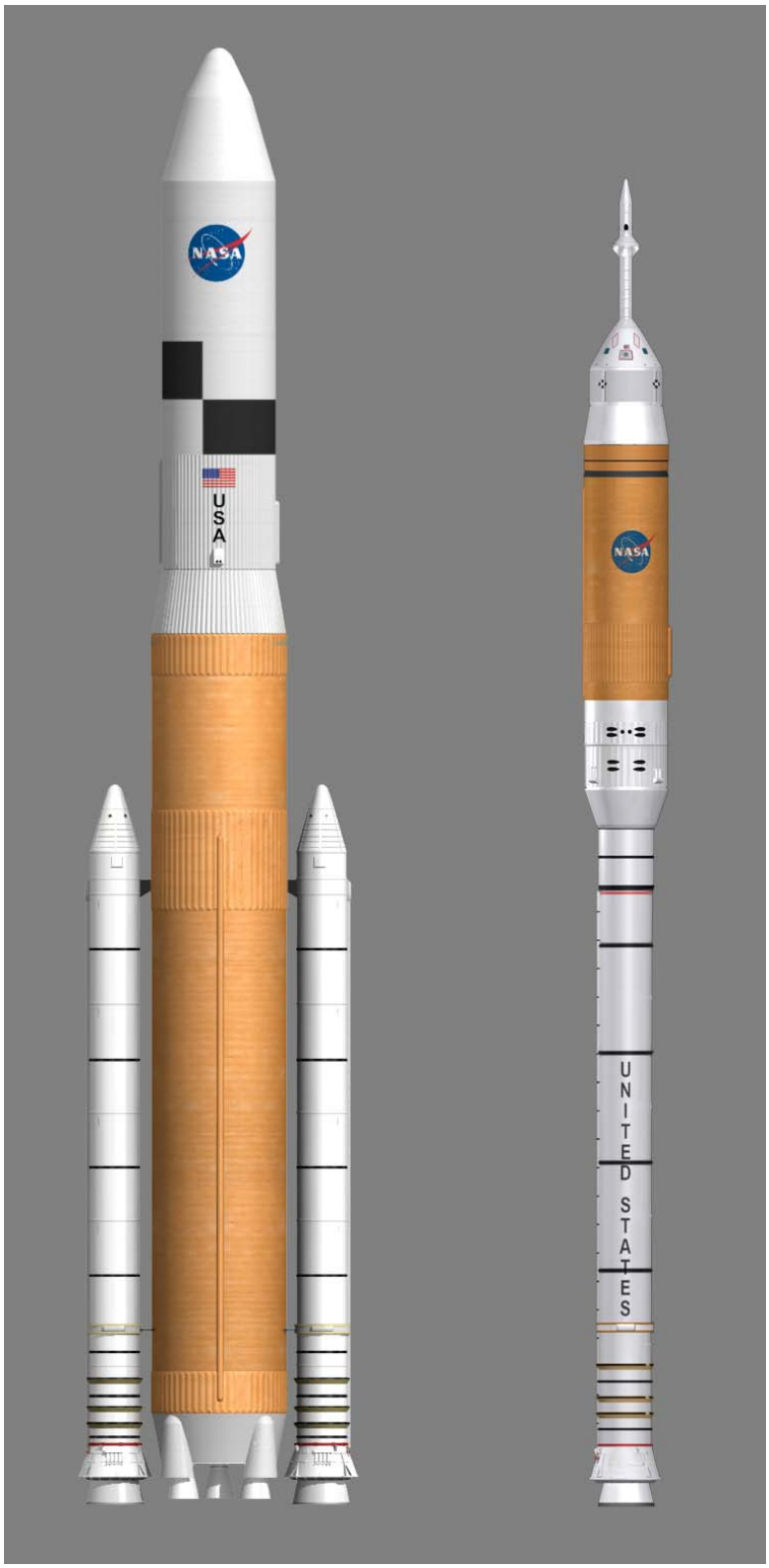


Figure A-10: Ares 5 and Ares 1 Rockets (Courtesy NASA)

Figure A-11: Earth Depart Stage, Orion & LSAM In Low Earth Orbit (Courtesy NASA)

Since the round-trip Δv requirements for travel between low lunar orbit and the lunar base are 3.4 km/sec and the N_2H_2/N_2O_4 rocket fuel chosen for the LSAM develops only about 3.0 km/sec exhaust speed, NASA has decided the LSAM will have two rocket stages, with one left behind on the lunar surface. This configuration copies the one used in the Apollo project. However, although NASA has no plans for this, if large quantities of water-ice can be harvested near the lunar base then the Earth/Luna transportation system could be considerably improved. A LH_2/LOX rocket with an exhaust velocity of 4.3 km/sec could provide 3.4 km/sec in a single stage, so a SSTO reusable rocket vehicle could be built to shuttle round-trip between the lunar base and low lunar orbit, refueling between trips at the lunar base. One such SSTO vehicle would be adequate to ferry to and from the lunar surface all visiting crew members on many sequential missions. This would eliminate the need to launch yet another fueled expendable LSAM for each expedition, reducing the mass launched from Earth per lunar visitor by more than 50%.

Although rocket technology has not advanced much since designs of the 1970s, improvements over the past 35 years in electronic technologies for communications, computation and control are exploited in NASA's new designs. The design scheme for Constellation relies on the now well-established capability to precisely navigate and remotely position multiple space vehicles to identical locations, to link and unlink them, and to transfer people and cargo between them. As with Apollo this approach requires a rendezvous linkup of the lunar lander and the Orion crew vehicle in lunar orbit, but precision landings of multiple vehicles at a pre-existing base location on the lunar surface is an utterly new requirement well beyond Apollo's capabilities. Another new Constellation feature is that the rocket systems injecting these modules into lunar transfer orbits will themselves be assembled in low Earth orbit from multiple separately launched components.

The Constellation project envisions two new expendable rockets code named "Ares 1" and "Ares 5", as depicted in Figure A-10. The Ares 1 would not be of heavy lift class but the Ares 5, with capabilities similar to the Saturn 5, would serve as the new heavy lift cargo booster. Together, the two rockets would accommodate missions to the ISS, Luna, and it is hoped, also to Mars. At present, conceptual design activities for the two Ares rockets are well under way at NASA's Johnson and Marshall spaceflight centers. NASA's present plan is that after Shuttle is retired, thus halting its stream of Shuttle operations expenditures, that continuing US government funding at the same ammual level can be used to rapidly develop the Ares rockets, These are to be derived from Shuttle components by making small design modifications. Plans are to build the new expendable Ares rockets at existing production facilities of the same Thiokol and Michoud contractors now producing Shuttle's solid boosters and Shuttle's external tanks.

The new Ares 1 rocket is a two-stage stacked rocket system. As shown in Figure A-10 it is topped with Orion modules and its launch abort system. The Ares 1 first stage is a single solid booster about 133 feet long derived from the two side-mounted boosters of Shuttle. After it has burned for 128 seconds to power the low speed accelerating ascent through the lower atmosphere, the first stage separates and lands in the Atlantic ocean via

parachute to be recovered by ship and refurbished for reuse (as is standard practice for Shuttle boosters). The Ares 1 second stage is a single J2X engine burning LH_2/LOX which provides most of the energy needed to reach orbit. As described on NASA's web sites, "The J-2X is an evolved variation of two historic predecessors: the powerful J-2 engine that propelled the Apollo-era Saturn IB and Saturn V rockets, and the J-2S, a simplified version of the J-2 developed and tested in the early 1970s but never flown". The Ares 1 will be able to launch a 55,000 lbm=25 tonne Orion module set with a 6 person crew and supplies from the Earth's surface into the circular low Earth orbit of the ISS. It also will likely be used to launch assorted small cargo modules for one-way delivery to the ISS or to the surfaces of Luna or Mars.

Ares 5 which stands approximately 360 feet tall is also a two-stage stacked rocket system but its first stage employs multiple rockets operating together in parallel. It is shown in Figure A-10 topped by a shroud enclosing either lunar modules or another cargo payload. The first stage will use five RS-68 engines burning LH_2/LOX mounted below a larger modified version of the space shuttle's external tank, with two solid propellant rocket boosters mounted on the tank's sides as they are on Shuttle. After a slow but accelerating ascent through the lower atmosphere the solid boosters will separate and land via parachute in the Atlantic Ocean not far from the Kennedy space center launch site to be recovered by ship and refurbished for eventual reuse. The RS-68 engines will continue to accelerate until an unstable orbit is reached, nominally 200 km altitude at apogee but only 50 km at perigee. The first stage then separates and reenters the atmosphere, burning up over the Pacific Ocean while the upper stage rocket fires briefly to circularize the orbit at a 200 km stable altitude. The upper stage of Ares 5, named the "Earth Depart" stage, will use the same J-2X engine employed in the Ares 1.

NASA documents state the Ares 5 rocket will be able to launch a 290,000 lbm (= 131.8 tonne) payload from Earth into a circular low Earth orbit. For lunar missions a substantial part of that 131.8 tonne LEO payload would be in the form of additional LH_2/LOX fuel for use by the Earth Depart stage to place lunar payloads into a trans-lunar injection trajectory. **NASA's plans for lunar missions call for tandem launches into the same low Earth orbit of both an Ares 1 rocket carrying an Orion module with a crew of 4 and a crewless Ares 5 rocket carrying the remaining modules as cargo together with fuel.** The two would link, then the Earth Depart Stage, would burn again, launching the resulting 66.9 tonne payload into a transfer orbit to Luna (see Fig. A-11).

Alternatively using a single Ares 5 rocket to launch cargo from Earth, NASA states that 54.6 tonnes could be placed into a trans-lunar injection trajectory. (Note that NASA's 131.8 and 54.6 tonne figures match Eq. A-12 for a full-to-empty mass ratio of 10, an exhaust speed of u=4.3 km/sec, and a speed increment Δv=11.2-7.98=3.22 km/sec.)

Another cargo possibility is to use multiple Ares 5 and/or Ares 1 launches into low Earth orbit, with one carrying a single large payload and the others carrying LH_2/LOX fuel. For instance, by using two Ares 5 launches and one Ares 1 launch, a 131.8-tonne payload

which had been manufactured as a single module and launched from Earth without separating it into subassemblies could be placed in a trans-lunar injection trajectory.

Cargoes placed into trans-lunar injection trajectories must include rockets and fuel to enter low lunar orbit, and even more additional fuel if the cargoes are to be soft-landed on the lunar surface. Assuming the decelerating rocket supplies a Δv of 2.38 km/sec matching Luna's escape speed, has a full-to-empty mass ratio of 10 and burns N_2H_2/N_2O_4 developing a 3.0 km/sec exhaust speed, Eq. A-12 predicts a maximum of only 39% of the mass placed into trans-lunar injection orbit could be soft-landed on the lunar surface.

As we shall see, the rocket Δv requirements to enter a well-timed transfer orbit to Mars are only slightly more than the Δv requirements to enter a transfer orbit to Luna, although the arrival speed at Mars is much higher. However, in contrast to the purely rocket-powered deceleration of lunar missions, payloads at Mars would use aerodynamic friction with the martian atmosphere. Aerodynamic friction has provided most of the deceleration for the six unmanned craft which have successfully landed on Mars. Large fractions of their entering masses were soft-landed using hypersonic aerodynamic braking followed by Disk-Gap-Band supersonic parachutes then by touch-down rockets. More engineering development will be needed for the one hundred times more massive payloads of manned missions, but aerodynamic friction will certainly serve. The new Ares rockets may even be able to soft-land more massive single payloads on Mars than on Luna, either as unmanned cargoes or as crewed vehicles.

Plans for rockets and associated equipment to use in manned Mars missions are not yet firm; indeed, they are completely open. Travel durations will be far longer than the durations of Earth/Luna transfers so the habitable cabin volume per crew member must not be too small. Two atmospheric entries will be required, one at Mars and one at Earth, so a single ablative heat shield destroyed in one entry would not be adequate. However, if the Constellation design philosophy succeeds for Luna it will likely be followed for Mars. Launches from Earth's surface using multiple Ares rockets will deliver several modules to low Earth orbit over time. They will be assembled there into larger linked configurations, then boosted into interplanetary trajectories during launch windows.

Constellation's multiple launch and multiple payload design philosophy will require that our skills in guiding remote payloads be extended to include martian aeroentries followed by precision landings at a base camp on Mars. In the past only Shuttle has achieved such precision landings after aeroentry. The need for precision landings on Mars will probably require some form of steerable hypersonic lifting bodies along with touchdown rockets.

A.5 Earth/Mars Orbital Transfers

A.5.1 Simplified Orbital Dynamics (Circular Planetary Orbits)

Broad schedular windows for launching into low Earth parking orbits preparatory for entering trans-lunar injection trajectories occur twice daily, and the subsequent Δv

requirements for lunar missions are almost constant throughout the monthly lunar orbital period. Astronauts can leave Earth for Luna or return from Luna to Earth on any day they choose, and the travel time each way is less than 5 days. However, the situation is very different for Mars. **For transfers between Earth and Mars the Δv requirements vary enormously as a function of time, thus creating narrow schedular windows far too important to be ignored.**

The simplest approximation of Earth's and Mars' orbits models them as coplanar circles with radii equal to their actual average distances from the sun, as shown in Figure A-12. Both planets orbit the sun in counterclockwise directions in this view from the North. This approximation is not an enormous distortion of reality since the actual elliptical orbits of Earth and Mars have small eccentricities and are in planes inclined to one another by only a very small angle (i.e., about 1.8 degrees). In this approximation it is easy to calculate the interplanetary separation distance versus time, as in Figure A-13.

Since in this circular orbit approximation the motion of each planet around the sun is uniform with speeds equal to the average speeds of their actual elliptical orbits, and since this approximation also preserves their orbital periods of 365.242 and 686.973 days respectively, it predicts that Earth should pass Mars in closest approach *opposition events* recurring regularly with a synodic period of precisely

$((365.242)^{-1}-(686.973)^{-1})^{-1}= 779.879$ days.

"Conjunction events" of maximal Mars/Earth distances also recur with the same period.

A.5.2 Hohmann's Optimal Transfer Between Coplanar Circular Orbits

In 1925, Walter Hohmann discovered that the minimum energy orbital transfer maneuver by rocket between circular coplanar orbits whose size ratio is not too large is an ellipse tangent to both orbits, as depicted in Figure A-14. With orbits of Earth and Mars modeled as coplanar circles, the Hohmann transfer orbits each have the following semimajor axis and **full** orbital period:

$$a_{\text{xfer}} = \frac{a_{\text{Earth}} + a_{\text{Mars}}}{2} = \frac{149{,}597{,}890\text{ km} + 227{,}936{,}640\text{ km}}{2} = 188{,}767{,}265\text{ km}$$

$$T_{\text{xfer}} = \left(\frac{(T_{\text{Earth}})^{2/3} + (T_{\text{Mars}})^{2/3}}{2} \right)^{3/2} = \left(\frac{(365.242\text{ days})^{2/3} + (686.973\text{ days})^{2/3}}{2} \right)^{3/2} = 517.72\text{ days}$$

Hohmann transfers in either direction straddle opposition events. An Earth-to-Mars Hohmann transfer departs (686.973-517.72)/(686.973-365.242)*365.242/2=96.07 days before opposition and arrives 517.72/2-96.07=162.79 days after opposition. A Mars-to-Earth Hohmann transfer departs (517.72-365.242)/(686.973-365.242)*686.973/2=162.79 days before opposition and arrives 517.72/2-162.79=96.07 days after opposition. In the Figure A-14 diagram, planet positions at an arbitrary opposition event are indicated by small "o" symbols connected by a radial line segment aligned with the sun. With all

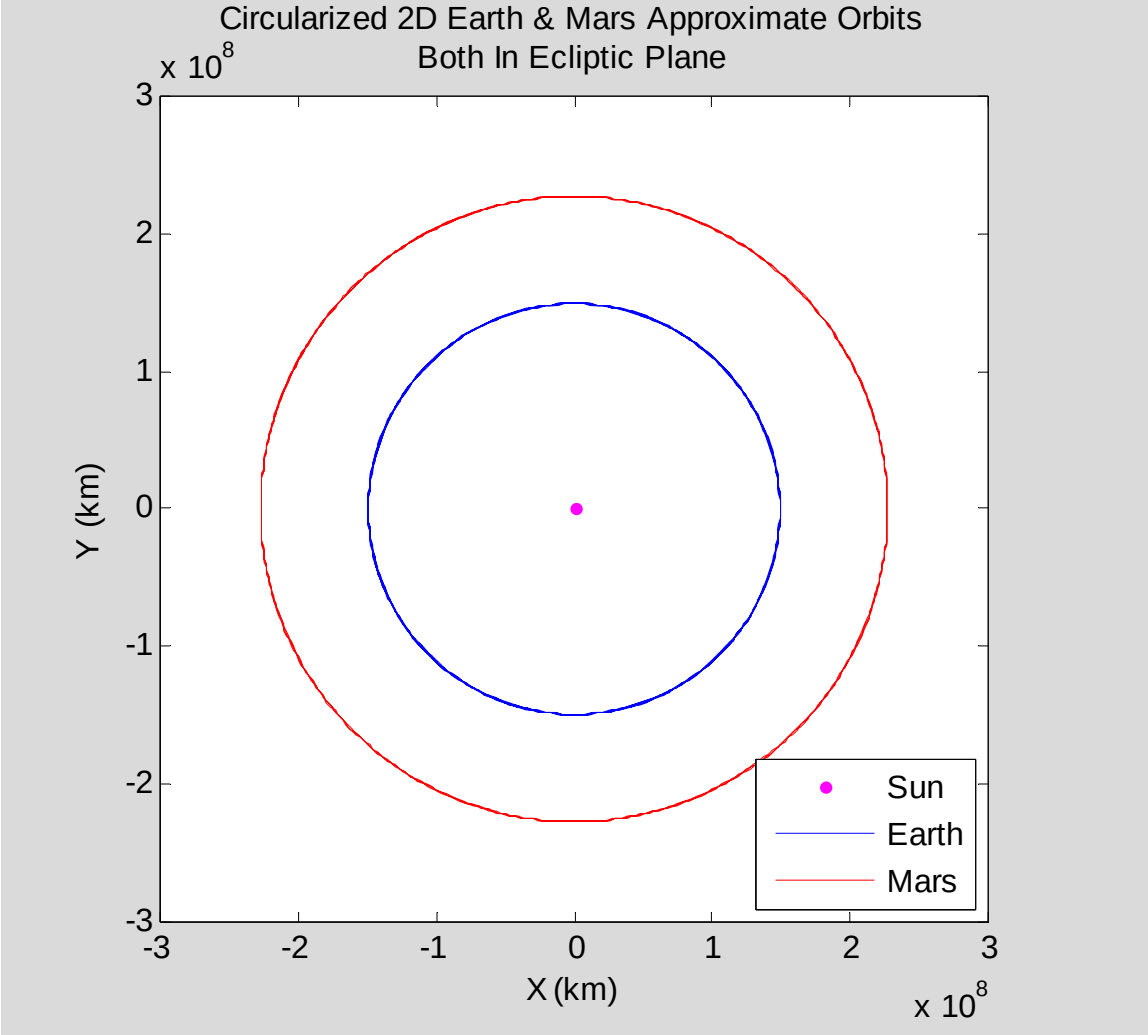


Figure A-12: Circular Coplanar Orbit Approximation

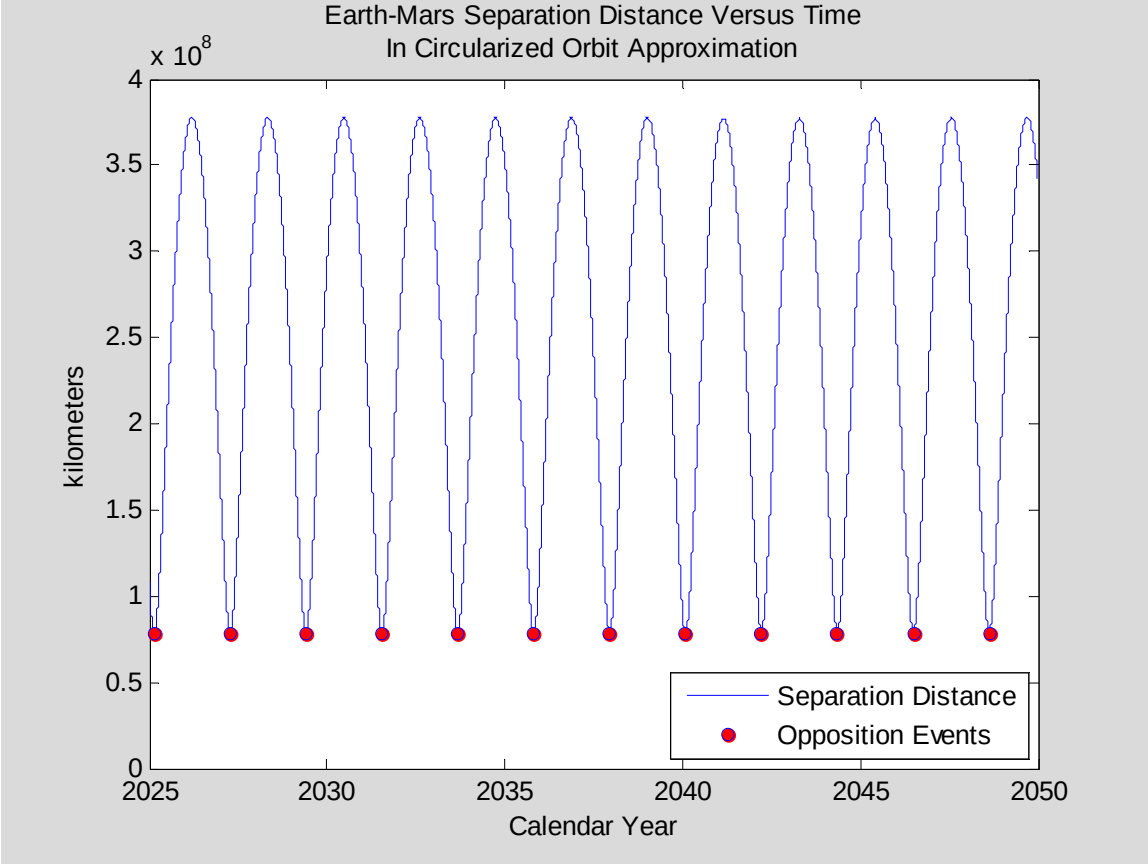


Figure A-13: Earth-Mars Separation Distance Vs Time (Circular Orbit Approximation)

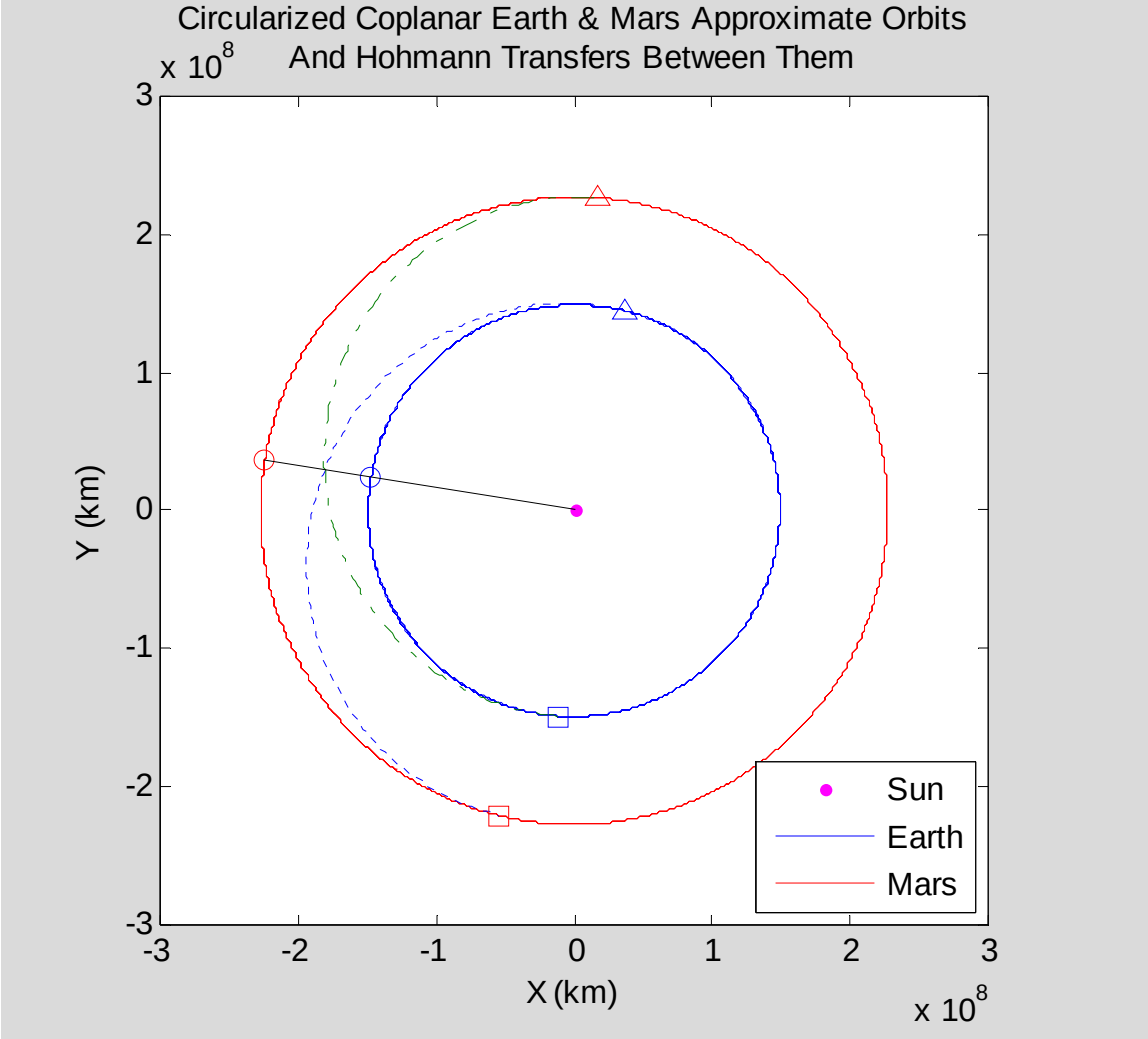


Figure A-14: Hohmann Transfers Between Circularized Orbits of Earth and Mars

motions counterclockwise about the sun (as seen from the North), planetary positions at departures are denoted by triangles while positions at arrival are denoted by squares. Transit times are each 258.86 days. In this approximation, a vehicle leaving Earth during one Hohmann transfer opportunity and returning during the next would be away from Earth for a total time duration of 779.879+258.86=1038.739 days with 779.879-258.86=521.019 days spent on Mars. This is termed a "Conjunction Mission" since it includes one conjunction event between two oppositions.

Hohmann transfer orbit rocket requirements are easily calculated from Kepler's laws. The transfer orbits' speeds at Earth's and Mars' distances from the sun are as follows:

$$v_{\text{xfer@Earth}} = 2\pi \frac{a_{\text{xfer}}}{T_{\text{xfer}}} \sqrt{\frac{a_{\text{Mars}}}{a_{\text{Earth}}}} = 2\pi \frac{188{,}767{,}265\,\text{km}}{517.724\,\text{days}} \sqrt{1.5237} \left(\frac{1\,\text{days}}{86400\,\text{s}} \right) = 32.74\,\text{km/s}$$
$$v_{\text{xfer@Mars}} = 2\pi \frac{a_{\text{xfer}}}{T_{\text{xfer}}} \sqrt{\frac{a_{\text{Earth}}}{a_{\text{Mars}}}} = 2\pi \frac{188{,}767{,}265\,\text{km}}{517.724\,\text{days}} \sqrt{\frac{1}{1.5237}} \left(\frac{1 day}{86400 s} \right) = 21.48\,\text{km/s} \quad \text{(A-16)}$$

Thus, the two elliptical Hohmann transfer orbits each have a counterclockwise speed with respect to the sun of 32.74 km/sec at the circularized radius of Earth's orbit and 21.48 km/sec at the circularized radius of Mars orbit. Since the transfer orbits are tangent to the circular planetary orbits where they meet, the velocity increments between transfer and planetary orbits can be calculated as simple speed differences. Transfer orbit speeds differ from the planets' mean heliocentric orbital speeds (see Table A-2) by 32.74-29.78=1.96 km/sec at Earth's orbit and by 24.1A-21.48=2.65 km/sec at Mars' orbit. Since it is also necessary to overcome planetary gravity at Earth and Mars, the total Δv requirements must also take into account the planetary escape velocities of respectively 11.19 km/sec and 5.03 km/sec. Assuming that all non-gravitational forces act only at low altitudes, conservation of energy may be used to combine planetary escape velocities with velocity increments between planetary and transfer orbits as the square root of the sums of their squares. This calculation yields the total Δv to be supplied by rocket or absorbed by aerobraking in ideal Hohmann transfers, as follows:

$$\Delta v^{(\text{Escape+Hohmann})}_{\text{Earth to Mars}} \approx \sqrt{(11.19)^2 + (1.96)^2} = 11.36\,\text{km/sec}$$
$$\Delta v^{(\text{Escape+Hohmann})}_{\text{Mars to Earth}} \approx \sqrt{(5.03)^2 + (2.65)^2} = 5.69\,\text{km/sec} \quad \text{(A-17)}$$

Thus, using Hohmann trajectories a velocity increment of 11.36 km/sec must be supplied by rocket to leave Earth's surface in a ballistic trajectory impacting Mars. Upon arrival at Mars, a 5.69 km/sec relative arrival speed increment must be absorbed by aerobraking. For the later return trip a velocity increment of 5.69 km/sec must be supplied by rocket to leave Mars' surface in a ballistic trajectory back to Earth. Upon arrival at Earth, 11.36 km/sec must be absorbed by passive aerobraking. Thus, the entire round trip involves a total calculated velocity change of 34.1 km/sec beyond the effects of gravity, of which 17.05 km/sec is accomplished by rocket and 17.05 km/sec by passive aerobraking. Actual Δv requirements would be slightly higher than these estimates because of additional atmospheric drag and gravity "launch losses" in each of the two planetary

launches. If we assume launch losses are about 1 km/s at each planet they would increase rocket requirements to about Δv =19 km/sec for a round-trip mission. Thus, an idealized Hohmann round-trip mission to Mars greatly exceeds the total Δv for an Apollo lunar mission (which was a weighted average of 12.4 km/s and 16.0 km/s plus launch losses).

A.5.3 Suboptimal (Non-Hohmann) Transfers Between Circularized Orbits

Locations of Earth and Mars in their circularized approximate orbits can be readily calculated for any arbitrary departure and arrival dates, then a heliocentric ballistic intercept trajectory linking them can be calculated. In general there are exactly two such ballistic trajectories which meet the boundary conditions, the Direct transfer going counterclockwise around the sun in the same direction as the planets, and the Retrograde transfer which travels clockwise. Direct transfers usually require less incremental Δv. Ballistic trajectories between Earth and Mars approximate heliocentric orbits shaped as ellipses, or as parabolas or hyperbolas if vehicle kinetic energy is sufficiently high. This heliocentric conic section approximation is accurate wherever solar gravity dominates, which is everywhere except near the transfer trajectories' ends where planetary gravitational fields dominate. The "patched conic approximation" used in Appendix B to evaluate Earth/Mars transfer trajectories effectively switches on and off solar and planetary gravity fields as the spacecraft passes through planetary "sphere of influence" boundaries, using Laplace's recommended "sphere of influence" sizes to minimize the error of this approximation. For Earth/Mars transfers this inclusion of planetary "spheres of influence" turns out to have only a small effect on mission schedule planning, shifting the rocket Δv function of transfer times by only 2 to 4 days at most. The Figure A-15 MATLAB 3D graphs show results of Appendix B's computation of many such transfer trajectories, with terminal dates at Earth and at Mars separately incremented on a 2D grid with 5 day resolution. Rocket Δv requirements were calculated by vector subtraction of the departure planet's velocity from the initial velocity of each calculated heliocentric transfer trajectory, then the departure planet's escape speed was also included as done in Eq. (A-17). Rocket Δv requirements approach infinity as the terminal times approach each other, thus dividing the computed Δv surface function into two separate regions, one for travel from Earth to Mars and the other for travel from Mars to Earth.

It is easy to see in Fig. A-15 the overall periodic structure of this Δv function which exactly repeats itself with the synodic period of 779.879 ≈ 780 days ≈ 2.1352 earth-years. However, it is difficult to extract detailed numerical information from Figure A-15, so a contour plot of two periods of this function along with labels and annotations is provided as Figure A-16. Because of the periodically repeating nature of the Δv function in this circular planetary orbit approximation, the Figure A-16 contour plot is universally valid (within the limits of that approximation) for all Earth/Mars transfers. Local minima were found by comparing calculated Δv array entries with their neighbors withoutinterpolation. Minimum rocket Δv values found were 5.72 km/s for Mars-to-Earth transit and 11.56 km/s for Earth-to-Mars, approximating Hohmann transfer values.

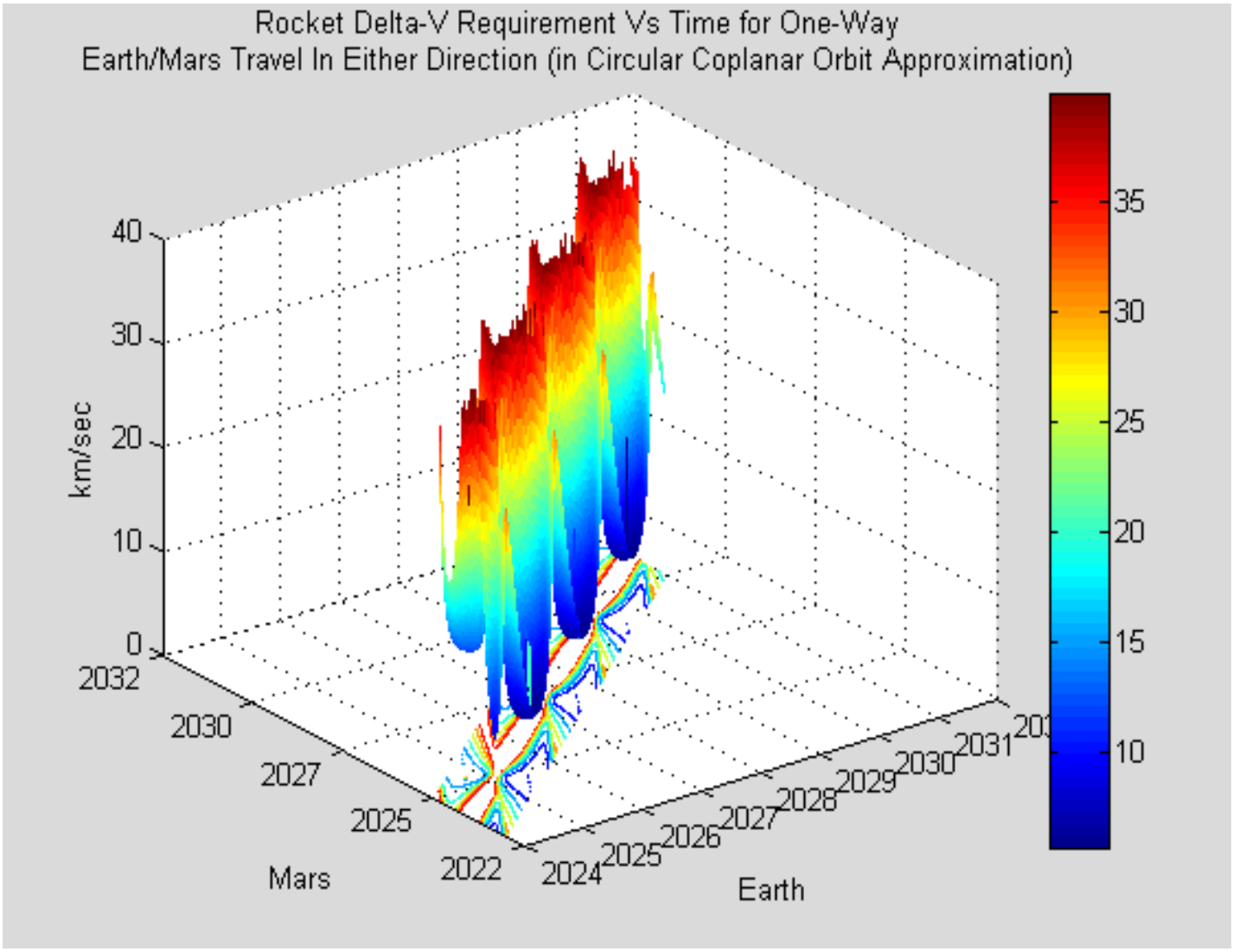


Figure A-15: 3D Plot of Earth/Mars Rocket ΔV vs. Terminal Times (Circular Coplanar Orbit Approximation)

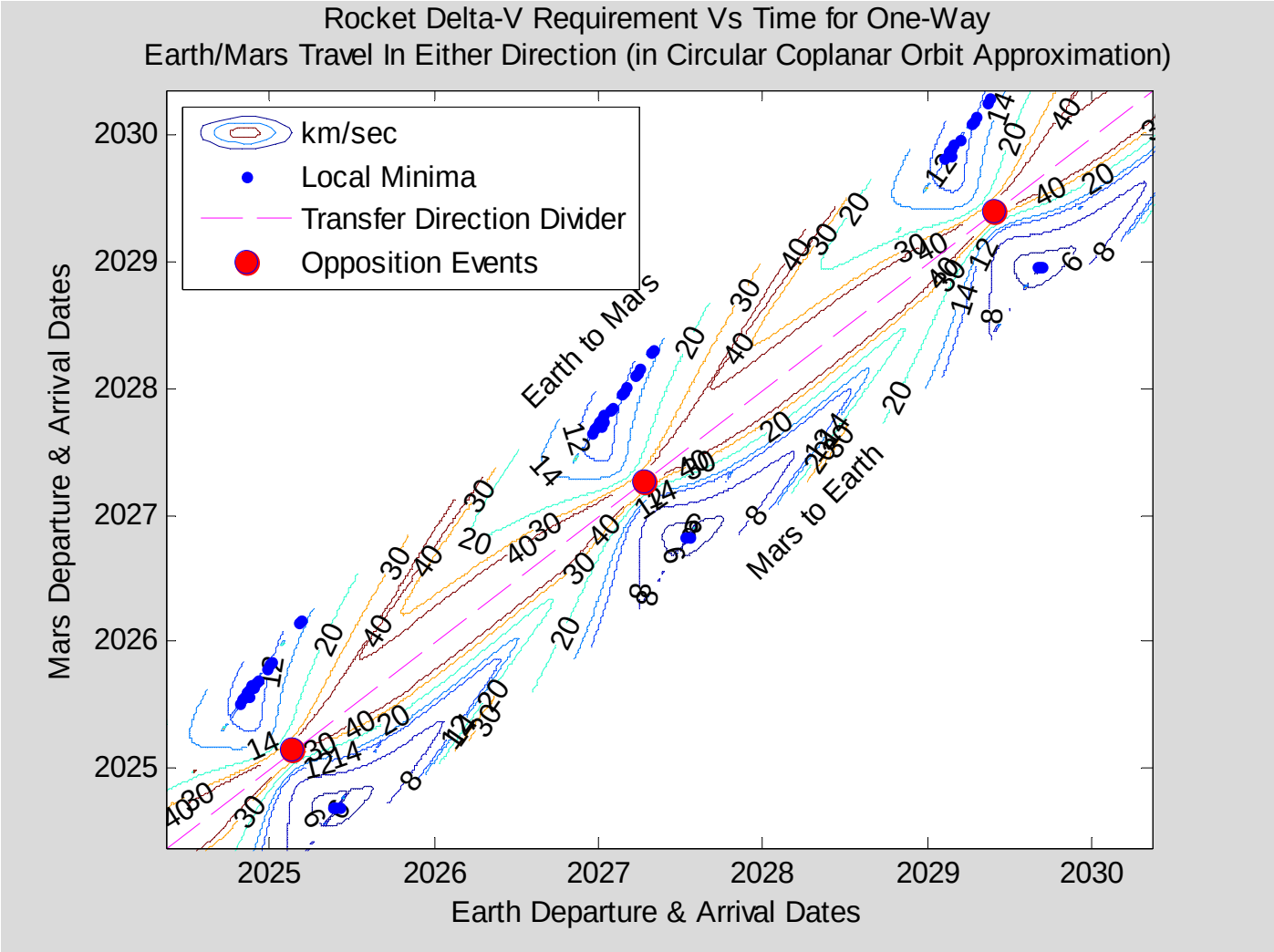


Figure A-16: Universal Rocket Δv Function for Earth/Mars Travel vs. Terminal Times
(in Circular Coplanar Planetary Orbits Approximation)

A.5.4 Trades Between Faster Earth-Mars Transfers Vs. Cost

An important issue to consider is the trade-off between Δv requirements and shortened transit times. What does it cost to reduce the time spent traveling between Earth and Mars? Reducing transit time below that of a Hohmann transfer requires increasing the Δv to be supplied by rocket. The tradeoff function, obtained by finding row minima of the 2D arrays containing calculated rocket Δv requirement as plotted in Figures A-15 and A-16, is itself plotted in Figure A-17. Because of the way this curve was calculated, minimum transit durations from Mars to Earth for different Δv values are shown as negative numbers while minimal transit durations from Earth to Mars are shown as positive numbers. The transit durations plotted range down to about 30 days and up to the 258 days of Hohmann transfer times.

To turn these calculated ΔV curves into cost curves we assume that a rocket system has already been provided to move a particular value of payload mass between Earth and Mars in a Hohmann transfer orbit. Any additional ΔV used to produce a faster transfer must be paid for by ***reducing payload mass*** and adding another rocket with a mass equal to the payload mass reduction. The fraction of the Hohmann transfer payload that would reach the destination after a shortened transit was calculated using Eq.(A-12) for different rocket fuels, assuming the additional rocket is single-stage and obeys $M_{full}/M_{empty}=10$. The resulting trade-off curves are plotted in Figure A-18 vs transit time. Figure A-18 predicts that by paying mass penalties of only 4%, the duration of a LH_2/LOX-fueled transfer from Earth to Mars can be reduced from 8 months to 6 months. An identical reduction of transfer duration for the return trip costs an 11% payload mass penalty if LH_2/LOX is the fuel. However, these are not very significant changes in timing compared to the overall mission duration which approaches three years. Larger reductions in transit time, e.g., to 4 months, require significantly larger sacrifices of payload (i.e., fewer supplies or equipment) and also may create aerodynamic problems due to their higher atmospheric entry speeds. Thus, manned missions to Mars will not deviate much from the basic Conjunction mission schedule. Unmanned missions delivering cargo will choose near-Hohmann trajectories, but manned missions may choose slightly faster free-return trajectory transfers whose additional costs are not too high.

Zubrin points out in his book The Case For Mars [Zubrin 1996, 83-84] that certain special Earth to Mars transfer trajectories straddling oppositions require about 6 months transfer time duration and also have the "free return" property that if they swerve slightly to avoid Mars their resulting path will eventually return them to Earth without additional rocket firings. Since these transfers only cost a 4% payload mass penalty over near-Hohmann transfers and would decrease time spent away from Earth by one month while increasing time at Mars by another month, they seem likely to be adopted notwithstanding the fact that their free returns to Earth would not occur until two years after leaving.

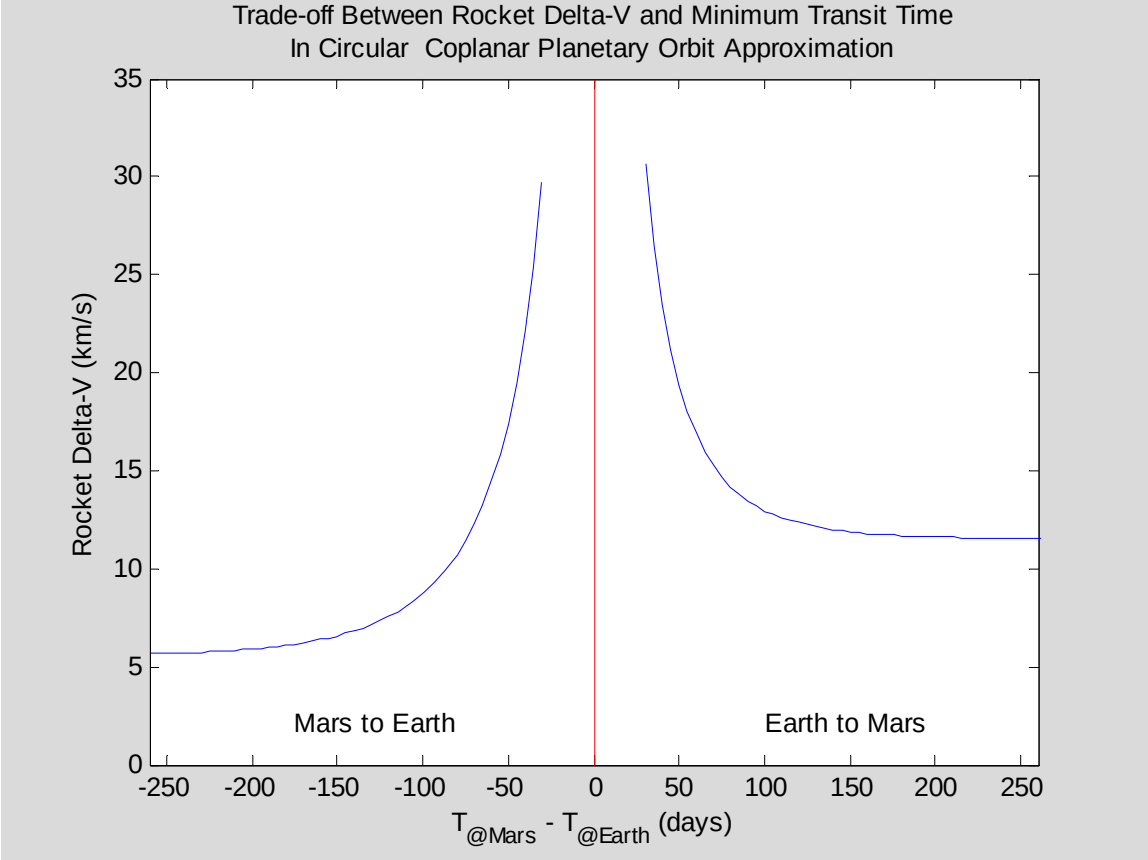


Figure A-17: Tradeoff Of Rocket ΔV vs. Earth/Mars Transit Time

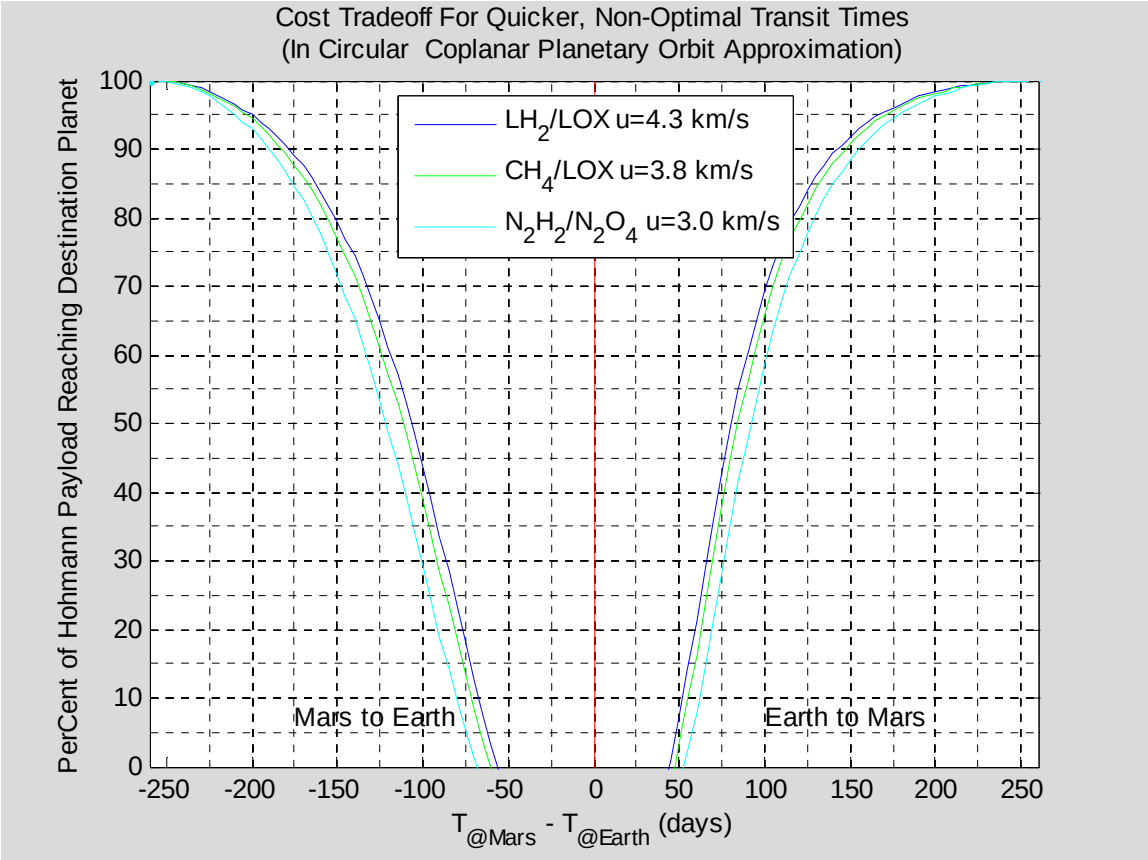


Figure A-18: Cost Tradeoff For Faster, Non-Optimal Earth/Mars Transfers (in Circular Coplanar Planetary Orbit Approximation)

A.5.5 Accurate Orbital Dynamics (Modeling Planetary Orbits As Ellipses)

Earth and Mars actually do not have precisely circular coplanar orbits, so ideal Hohmann transfers are not exactly possible. Earth actually circles the sun in a low eccentricity elliptical orbit in which its distance from the sun varies between 147.1 million kilometers and 152.1 million kilometers. Mars' circles the sun in a slightly more eccentric orbit in which its distance from the sun varies from 206.7 million kilometers to 249.3 million kilometers. The two orbits are not coplanar, being inclined to each other by 1.849736 degrees. Figure A-19 shows the XY projection of the orbits of Earth and Mars in XYZ heliocentric-ecliptic coordinates as seen from the north side of the ecliptic plane. Because the two orbital periods are not commensurate and their orbits are ellipses, the Earth-Mars distances for successive oppositions vary between 56 million kilometers and 97 million kilometers. The time intervals between successive oppositions also vary by a few days from their approximately 780 day average value. Figure A-20 plots actual interplanetary distance versus time, with distance minima opposition events marked.

As done for circular coplanar approximations of planetary orbits, the actual 3D position and velocity vectors of Earth and Mars can be calculated at any chosen departure and arrival times, however their actual orbit element parameters must be used along with appropriate formulae. Ballistic trajectories can be calculated between the resulting departure and arrival planet locations and times exactly as before. However, it turns out that calculated ballistic transfer trajectories sometimes require excessive velocity changes when the two planets' orbits are inclined even slightly with respect to one another. In such cases it helps greatly to introduce a small "plane change" rocket burn to take place in space far from the planets. The optimal combination of two ballistic trajectories addressing such orbital plane changes was published in 1963 [Fimple]. The Fimple maneuver can be optimal when the ballistic transfer plane defined by the location of the sun, the location of the departure planet at departure time and the location of the arrival planet at arrival time is inclined greatly with respect to either planet's orbital plane.

As was done for the circular orbit approximation, additional calculations in Appendix B determined rocket Δv requirements for true Earth/Mars transfers, also using the "patched conic" approximation but this time using actual planetary orbital parameters and adding Fimple maneuvers to the transfer types previously considered. The resulting Figure A-21 contour plot shows, as a function of departure and arrival dates from years 2025 to 2030, the total Δv to be supplied by rocket for Earth/Mars transfers in either direction. Each point on the contour plot represents a different transfer trajectory, chosen as the best of three transfer schemes, i.e., as the minimum Δv of Fimple, Direct and Retrograde transfer maneuvers. Launch losses are not included.

Comparing Figures A-21 and A-16 shows they are not very different. Successive periods in the more accurate Figure A-21 cannot precisely repeat but are very similar to each other and to the identical successive periods of Figure A-16. After surveying several cycles of this non- repeating function, it appears to me that minimal Δv Earth-to-Mars transfers don't vary more than about 0.5 km/sec between best and worst case oppositions.

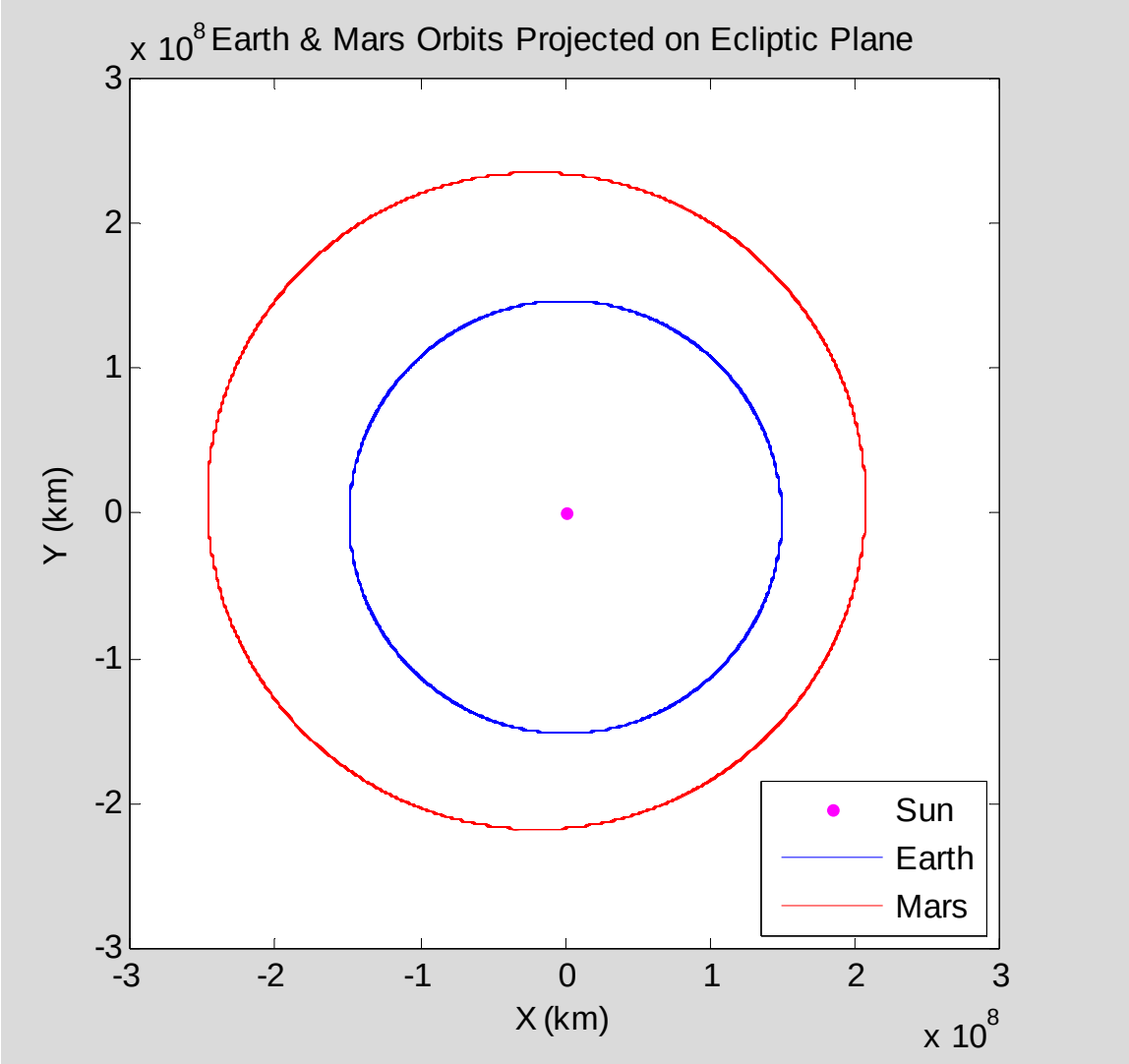


Figure A-19: Actual Earth & Mars Orbits Projected on Ecliptic Plane

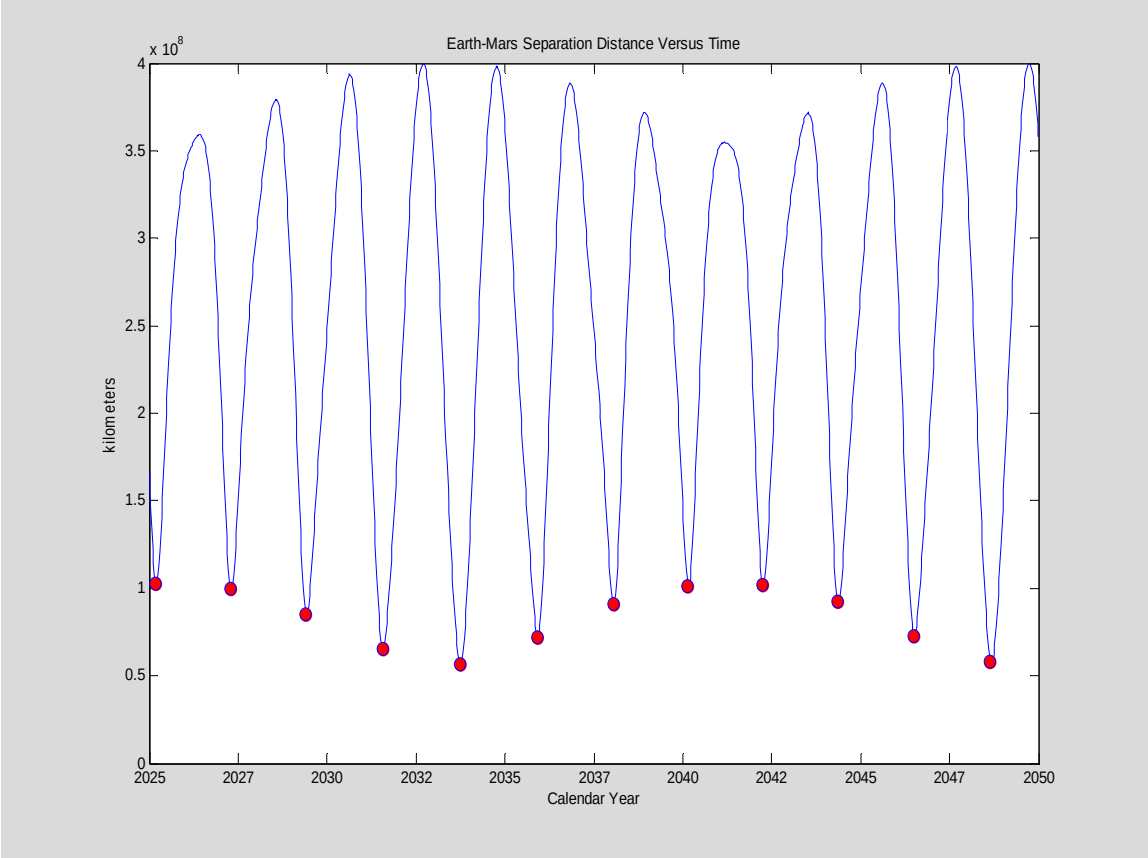


Figure A-20: Actual Earth-Mars Separation Distance vs Time

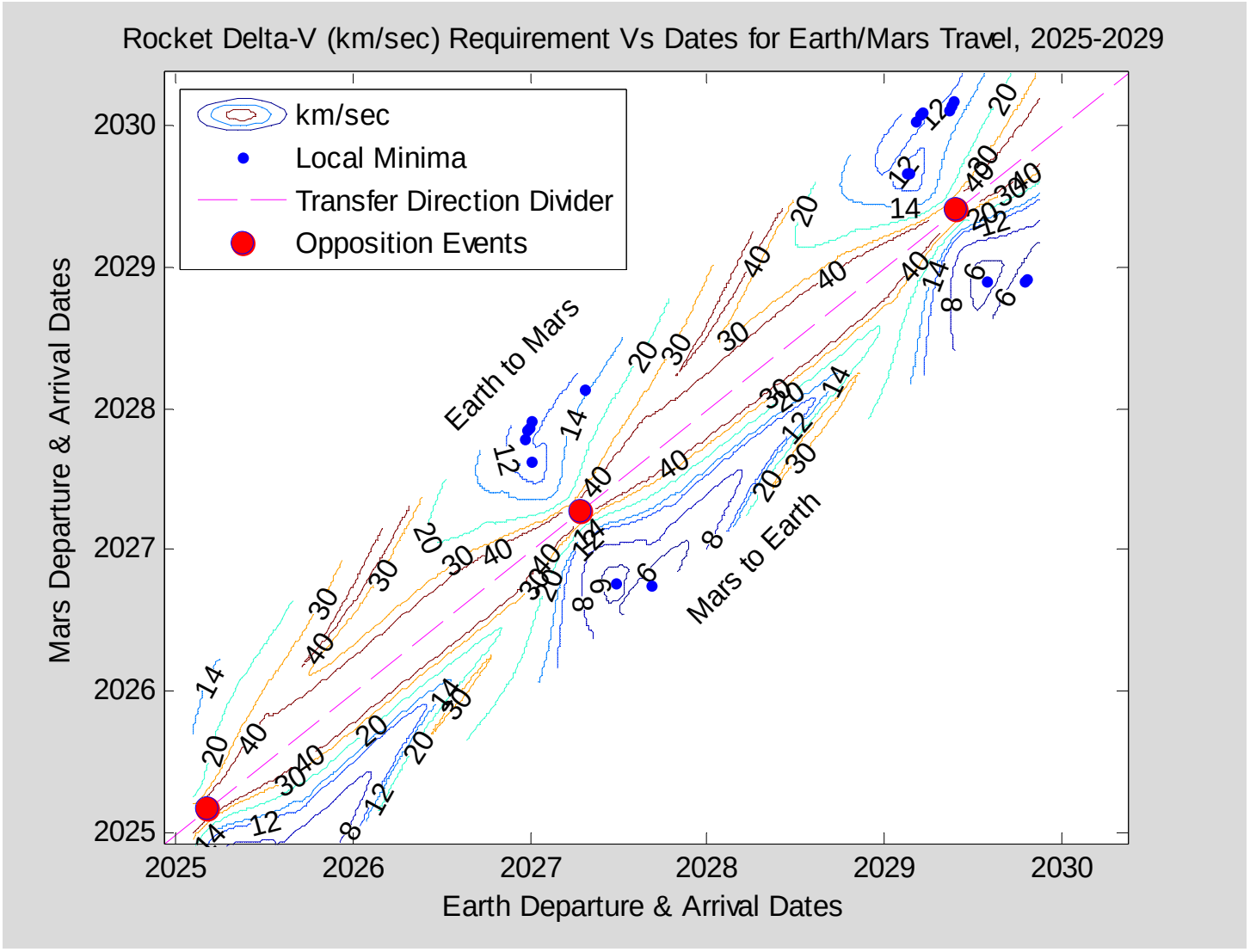


Figure A-21: Rocket ΔV vs Terminal Dates For Actual Earth/Mars Travel 2025-2029

A.5.6 Conjunction Mission Example Typical Of Future Manned Missions

Figure A-22 illustrates a particular 940-day roundtrip expedition from Earth to Mars including one conjunction in its middle, featuring minimum-Δv transfers in each direction straddling two successive oppositions The expedition leaves Earth's surface in a manned space vehicle on 04-January-2027. A multi-stage rocket developing a calculated Δv=11.58 km/sec launches the vehicle into a ballistic trajectory from Earth to Mars. After 225 days in space the vehicle tangentially enters Mars' atmosphere on 17-August-2027 at a speed of 6.94 km/sec. Protected by its heat shield, the vehicle slows by aerodynamic friction then lands near Mars' equator using a parachute and a touch-down rocket. The mission then spends the next 465 Earth-days, equivalent to 453 Mars-days or "sols", on Mars surface. It leaves Mars' surface in a return vehicle on 24-November-2028 when a rocket developing a calculated Δv=5.69 km/sec launches the vehicle into a ballistic trajectory from Mars to Earth. After 250 days in space the vehicle tangentially reenters Earth's atmosphere at a speed of 12.50 km/sec. Protected by a heat shield the vehicle slows by aerodynamic friction until its parachutes open, then lands in the ocean where it and its crew are recovered by ship. Total rocket ΔV for the mission is 17.27 km/sec plus launch losses, likely totaling about 19.3 km/sec.

A.5.7 Opposition Mission Example Demonstrates It Is Unattractive

If considerably more rocket fuel could be used then a shorter round-trip expedition to Mars, termed either a *single-opposition* mission or simply an *opposition* mission, would become feasible. Advocates of opposition missions feel a benefit exists in their reduced time away from Earth. In an opposition mission, the Earth departure, transit, and arrival at Mars all would occur immediately prior to an opposition event, and the Mars departure, transit, and arrival back at Earth all would occur immediately after the same opposition event. Figure A-23 illustrates such an expedition which lasts only 620 days. The vehicle leaves Earth at a speed of 20 km/sec on 28-Jun-2026, then coasts for 210 days until a 24-Jan-2027 landing on Mars. The mission stays on Mars' surface for 190 days. The vehicle leaves Mars on 02-Aug-2027 at 8 km/sec, then coasts for 220 days until a 09-March-2028 reentry and landing back on Earth, Total rocket Δv for the mission is 28 km/sec plus launch losses, likely totaling about 30 km/sec. With such large Δv values, variations could even involve flying by Venus in one of the travel legs in order to reduce time further.

A single-opposition mission does not appear to be a bargain for science. It spends about the same time in transit as a near-Hohmann mission, it costs 10.73 km/sec more Δv, but it has a negative science payoff, i.e., ***less*** time spent on Mars. **It seems likely that manned Mars missions for *scientific* purposes would never adopt such single-opposition missions.**

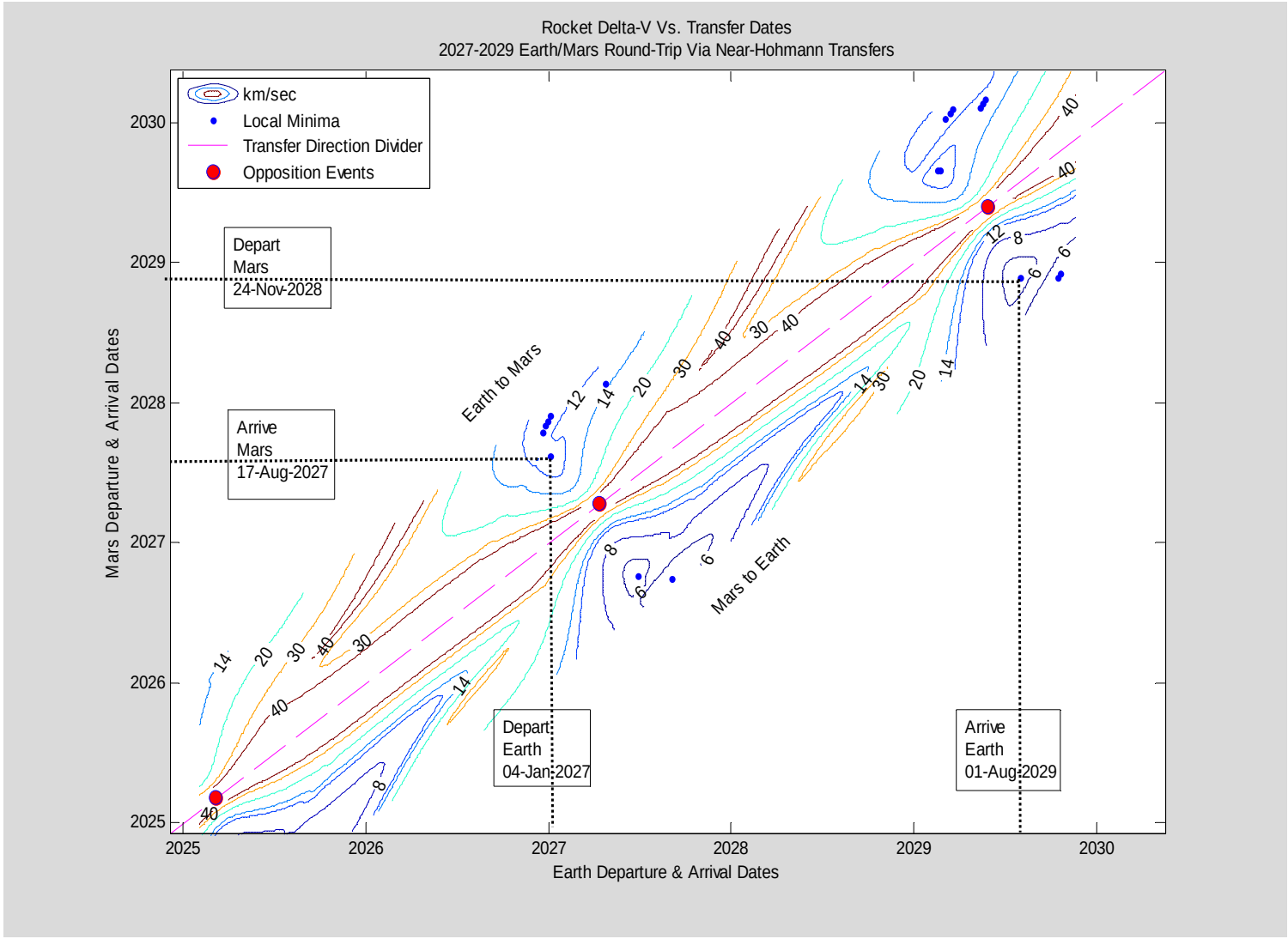


Figure A-22: A Conjunction Mission (With Transfers Straddling Two Oppositions)

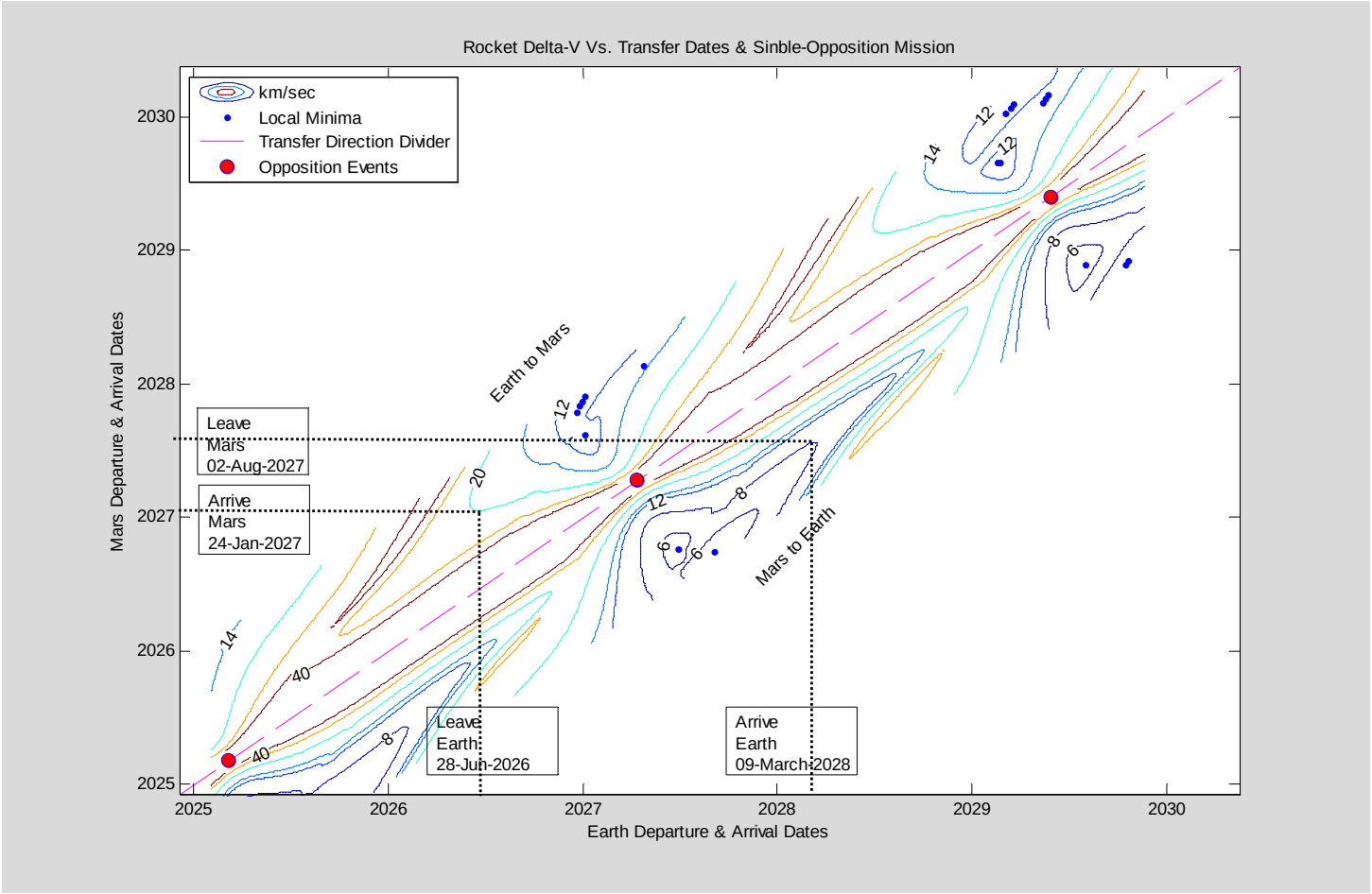


Figure A-23: A Round-Trip Single-Opposition-Mission To Mars

A.6 Mars Mission Schedules & Rocket Requirements Summary

In light of the above discussions it seems clear that manned Mars missions will adopt the conjunction mission scheme and will follow interplanetary transfer schedules straddling successive opposition events. They will use either 8 month near-Hohmann transfer trajectories or the slightly faster 6 month "free-return" trajectories which cost a small additional mass penalty. Each crew will be away from Earth for almost 3 years, leaving Earth 3 or 4 months before an opposition and arriving back at Earth 3 or 4 months after the next opposition. Each crew will spend between 17 and 20 months on Mars' surface.

The total rocket Δv needed to launch from Earth's surface into a near-Hohmann transfer trajectory to Mars is 11.5 to 12.0 km/sec plus about 1 km/sec launch losses. Another 0.5 km/sec would be required to shorten the transfer duration to 6 months. Equivalently, the rocket Δv *increment* needed to launch from a 200 km altitude low Earth orbit (LEO) into a transfer trajectory to Mars is 3.7 to 4.2 km/sec for a near-Hohmann transfer and up to 4.7 km/sec for a 6 month transfer. Using the Ares 5's Earth Depart stage or its equivalent to boost from LEO into the Earth-to-Mars transfer trajectory will therefore require a LEO mass of LH_2/LOX fuel in the range from 1.36 to 1.98 times the mass of the transferred payload as per Eq. A-5, assuming the rocket exhaust speed is 4.3 km/sec. For example, Eq. (A-12) predicts the 131.8 tonnes launched into LEO by a single Ares 5 vehicle could include LH_2/LOX fuel and a 34.4 tonne payload accelerated another 4.7 km/sec by the Earth Depart Stage to enter a 6 month transfer trajectory to Mars during a worst-case opposition. Larger payload masses would require multiple launches and joining in LEO. The rocket Δv needed to return from Mars' surface into a trajectory to Earth will be 5.5 to 6.5 km/sec plus launch losses. Equivalently, the Δv *increment* needed to return from a 200 km altitude low Mars orbit (LMO) into a transfer trajectory back to Earth will be 2.0 to 3.0 km/sec. If a LH_2/LOX rocket with 4.3 km/sec exhaust speed were used to return from LMO back to Earth, the fuel required would be in the range from 0.59 to 1.01 times the transferred payload, as per Eq. A-5. Aerodynamic friction with some minor assistance from rockets will be used to decelerate at each planet and to land.

Assuming a full-to-empty mass ratio of 10 and an exhaust speed of 4.3 km/sec, Eq. A-12 predicts that of the 131.8 tons launched into LEO by a single Ares 5 rocket, 47.3 tonnes of cargo could be sent to Mars in a near-Hohmann transfer during a good opposition while only 40.5 tonnes of cargo could be sent during an unfavorable opposition. Thus, a single Ares 5 can send 43.4±3.9 tonnes of cargo to Mars, depending on how the orbits line up for that particular opposition. A more massive single payload module up to 131.8 tonnes could be sent to Mars by first launching it into LEO then supplying additional fuel for its Earth Depart stage via additional launches. On the other hand, not all of the mass sent to Mars can be soft-landed on Mars' surface since some mass must be used for special equipment dedicated to slowing down, in a process termed Entry, Descent and Landing (EDL)..

A.7 Mars Entry, Descent and Landing (EDL) and Aerocapture

In every space flight in which a vehicle mass has returned to Earth, including all manned space flights, frictional aerodynamic heating in the Earth's atmosphere instead of rocket thrusting has absorbed essentially all the kinetic energy of the returning mass. Indeed, if this had not been the case and rocket powered deceleration had instead been necessary to slow returning spacecraft, the Δv requirements of all manned missions, including all missions to low Earth orbit or to Luna, would have doubled. With such prohibitively high rocket Δv requirements, manned spaceflight would never have been attempted.

However, aerodynamic entry braking does work well. The reentry heat is actually generated in the air (not in the vehicle's surface) by the compression of a hypersonic shock wave which precedes the spacecraft's leading surface by several centimeters, thus converting the shocked air to ionized plasma. In addition to manned payloads this phenomenon has been harnessed for military missile warheads and for scientific unmanned payloads entering Earth's or other bodies' atmospheres. It has been used for all landings on Venus and Mars and was recently used by the Huygens probe of the Cassini mission to Saturn which entered the thick atmosphere of Titan, Saturn's largest moon. Its most audacious application by far was in the Gallileo mission's descent probe which in 1995 entered Jupiter's atmosphere at the almost unimaginable speed of 47.1 km/sec, then slowed to subsonic speed and finally descended slowly under its deployed parachute as it telemetered its direct measurements of Jupiter's internal atmospheric composition.

The present technology for aerodynamic Entry Descent and Landing (EDL) on Mars was developed more than 35 years ago in preparation for the twin Viking Lander missions. The original Viking method used a conical heat shield to slow from interplanetary arrival hypersonic entry speeds of 7 km/sec to 0.7 km/sec. Special Disk-Gap-Band (DGB) supersonic parachutes then opened, further slowing the descent rate to less than 100 m/sec. Finally, a small "touchdown" rocket stage fired to achieve a soft landing on the surface. The Viking scheme was modified in a minor way for the Mars Pathfinder (MPF) mission which landed the Sojourner robot and for the twin Mars Exploration Rover (MER) missions which landed the Spirit and Opportunity robots. The MPF and MER modifications simply replaced part of the touchdown rockets' final actions with airbags. The scheme was used again in 2008 for the Mars Phoenix lander which again used touchdown rockets rather than airbags.

There is a consensus that EDL technology must be extended in preparation for manned Mars mission, so this is an active research area at present. The two EDL problems which are being addressed today by engineering research are that (1) the existing EDL methods do not scale up well to heavier payload mass regimes, and (2) the existing EDL methods do not provide pinpoint landings. The maximum payload among the six robotic craft which have so far successfully landed on Mars was less than 600 kg (i.e., 0.6 tonnes) but according to NASA's plans, manned missions will need to soft-land individual payloads between 20 tonnes and 100 tonnes. Furthermore, future landings will need pinpoint

precision with errors not exceeding about 10 meters in order to deliver equipment to a base, rather than the 100 km long 1-σ error ellipses typical of Viking EDL technology.

Publications report the mass scale-up problem starts with payloads greater than about 2 tonnes and arises from the increase in mass per unit aerodynamic surface area, both for the aeroshell which acts during the hypersonic phase and for the supersonic parachutes which act later. Possible technical solutions being investigated include the use in both hypersonic and supersonic phases of large light-weight ring-shaped inflatable devices called ballutes whose stiffness due to their internal compressed air would guarantee a large cross sectional area, first for greater hypersonic friction and later to support an extended parachute. The costly alternative would be an increased reliance on rocket fuel for EDL. This latter approach was followed in a 1998 NASA simulation study [Drake 1998] which concluded that to soft-land a single 40 tonne cargo on Mars' surface a total of 61.8 tonnes would need to be launched from Earth into a transfer orbit to Mars. A different study conducted at the Georgia Institute of Technology [Braun and Manning 2006] more pessimistically concluded that 100 tonnes would need to be launched into a transfer orbit to Mars to soft-land a single 35 tonne payload on Mars' surface. Thus, one expert group claims that t 2/3 of a large mass delivered to Mars' atmosphere can be soft-landed on the surface as payload while another expert group says the fraction is 1/3. This lack of agreement among technical experts attests to the need for their various computer simulations to be benchmarked against results from engineering technology experiments which NASA should organize and fund. Table A-5 shows calculated payload masses which could be soft-landed on Mars assuming different numbers of rocket launches to LEO per payload and assuming different EDL mass fractions.

The pinpoint landing issue arises because like Earth, Mars' atmospheric density is highly variable at the high altitudes at which hypersonic entry braking occurs. The only practical way to avoid having a large (100 km) dispersion-produced error ellipse in the landing location is to use a hypersonic lifting body like the Shuttle instead of a conical heat shield and to fly it like Shuttle using active computer-based feedback control during the entry phase. NASA/JPL is designing such systems at its DARTS laboratory and plans to test a resulting pinpoint landing system, code named the Mars Smart Lander (MSL) in their next large rover mission to Mars, presently scheduled for 2009. That mission will return engineering evaluation telemetry data which it is hoped will be useful in extending its new pinpoint landing capability to the more massive payloads that will be needed later.

Aerocapture is a frequently suggested and analyzed but never yet tried technique to slow an arriving interplanetary spacecraft without landing. In aerocapture, the hyperbolic arrival trajectory of the spacecraft is routed into the atmosphere in such a way that it slows in one pass to less than escape speed, then skips back out of the atmosphere captured in an elliptical orbit about the planet. The aerocapture scheme then uses a tiny rocket burn one-half orbit later at apoapsis in order to raise the elliptical orbit's periapsis out of the atmosphere. After pausing in the resulting parking orbit for an indefinite period, another small rocket burn relowers periapsis back into the atmosphere to initiate EDL for any portion of the spacecraft slated to land.

Table A-5: Calculated Ares5 Payload Capability For Soft-Landing On Mars

Assumed EDL Mass Fraction =	**1/3**	**1/2**	**2/3**
Number of Ares 5 Launches Per Soft-Landed Payload			
1	**14.5±1.3 tonnes**	**21.7±2.0 tonnes**	**28.9±2.6 tonnes**
2	**28.9±2.6 tonnes**	**43.4±3.9 tonnes**	**57.9±5.2 tonnes**
3	**43.4±3.9 tonnes**	**65.1±5.9 tonnes**	**86.8±7.8 tonnes**
4	**57.9±5.2 tonnes**	**86.8±7.8 tonnes**	**115.8±10.4 tonnes**
5	**72.3±6.5 tonnes**	**108.5±9.8 tonnes**	**144.6±13 tonnes**

The advantage of aerocapture is that compared to using rockets it provides a cheap way to transition between an incoming interplanetary trajectory and an orbit about a planet. NASA's planners of manned Mars missions have deemed it useful to include a pause in orbit before landing, e.g., perhaps to wait for a global dust storm to clear. Aerocapture would also reduce the aeroshell's heat load per atmospheric pass. But perhaps its most valuable feature is that it would permit leaving in Mars orbit any equipment which would not be useful on the Mars surface, especially any systems designed only for life-support operations in space or in zero gravity such as extra-light solar arrays or thermal radiators or an artificial gravity centrifuge. It would be wasteful to carry these masses down to Mars' surface only to relaunch them into orbit 18 months later. Instead, aerocapture allows them to be left orbiting Mars during a crew's stay on the martian surface.

The main impediment to using aerocapture has been the large uncertainty in predicting in advance the air density to be encountered during a hypersonic aerocapture pass through the upper atmosphere. If the air density were too high the spacecraft would land or burn up while if the density were too low the spacecraft would continue through interplanetary space without being captured by the planet's gravity. To make aerocapture reliably feasible it will be necessary to either obtain accurate atmospheric density profiles within a few hours before the aerocapture pass in order to fine-tune its entry altitude, or use a shaped aerodynamic body and actively feedback-control its flight orientation. Since either of these could in principle be done, the impediment will eventually be removed. NASA's plans to include aerocapture before landings in manned Mars missions seem realistic. Aerocapture will be standard practice before manned Mars missions begin.

A.8 Orbit Transfer Implications for Manned Mars Missions

I disagree with aspects of NASA's present vision for manned Mars missions. I hope they will evolve substantially and I believe they must. Planned sizes of crews and vehicles should be increased along with their reliance on reusable systems. Happily, technical details of NASA's Mars plans are not firm, nor should they be if manned Mars missions will not begin until well after 2020. Therefore, I will state here my own technical opinions about how manned Mars missions should, and hopefully will, be carried out.

Since the duration of each manned mission to Mars will be more than one hundred times as long as the longest of the Apollo lunar missions, a different approach is required for life support design, for crew selection, and for equipment choices. Mars missions will need provisions for personal hygiene and exercise entirely avoided by Apollo. More pressurized volume is needed while in space for 6 to 8 month trips, at least 20 m^3 per crew member (equivalent to a 10'-by-10' room with a 7' ceiling) but preferably more. Of course, *habitable* volume per crew member is considerably less than this pressurized volume because pressurized volume also contains supplies and equipment. Recycling equipment for water and air will be needed to avoid excessive consumables mass. The

crew transfer vehicle should also include a small well-shielded region for astronauts to retreat to during solar flare high radiation events lasting a day or two. Such a shielded region is readily provided in a larger spacecraft but is difficult in a small vehicle.

The likelihood of crew member illness or equipment failure during such missions lasting almost three years far from Earth's safety is significant, so each crew should include a medical doctor along with medical and dental equipment and engineers with appropriate tools for trouble-shooting and repairs. Since a larger skill set is required from the crew, a Mars mission should include more crew members than lunar missions. NASA's mission planners agreed that the originally proposed crew of 4 is too small for Mars but then only proposed a crew of 6 (which would land back on Earth using the Orion crew module being developed now for the ISS and Luna). My opinion is that a crew of 6 is still far too small to accomplish any significant scientific research in addition to the difficult maintenance and life support activities essential for their own survival. Maybe a crew of 6 could plant a flag and survive until it was time to return home, but they could not accomplish much for science. A crew of 15 highly cross-trained individuals might be adequate for each science mission but more (e.g., 30) would be far better.

It is appropriate to provide artificial gravity in the Earth/Mars transfer vehicle to counter the long-term effects of weightlessness. Many astronauts who have stayed in weightlessness for 6 months or more on the ISS or on the previous Russian space station, MIR, have needed help standing after returning to Earth's gravity or have even fainted due to blood pressure issues. There has also been permanent damage. These problems must be avoided on Mars where there are no people waiting to help astronauts readjust to gravity. Artificial gravity in the transfer vehicle would eliminate this problem.

Proposed artificial gravity schemes rotating either the entire crew transfer vehicle or the vehicle tethered to an external mass are both flawed since they would preclude lightweight solar arrays and thermal control radiators while also making external pointing for communications or solar power extremely difficult. A far better approach to artificial gravity would provide within part of a non-rotating crew transfer vehicle's internal pressurized volume a large inertially isolated rotating and self-balancing axisymmetric chamber which crew members could enter and exit at will through an axial portal. This scheme would continuously provide artificial gravity in one part of the pressurized crew cabin while weightlessness would prevail in the rest of the cabin. In addition to providing crew members with weight-bearing exercise, advantages of such a centrifuge include that it would simplify both washing for personal hygiene and toilet waste collection, it would enable the operation of water purification and recycling equipment which need gravity for distillation and other fluid separation schemes, and it would facilitate any necessary emergency medical procedures such as surgery. It is entirely feasible to include such an internal centrifuge within a conical or cylindrical crew transfer vehicle provided the vehicle's diameter is not less than about 10 meters.

The Ares 5 rocket's stated 131.8 tonne LEO capability allows it to orbit a large and massive pressure-tight Earth-manufactured crew vehicle for Earth/Mars transfers. I

suppose it could launch a payload as wide as **15 meters (i.e., 50 feet) in diameter**. In the Figure A-24 diagram, my proposed transfer vehicle subsumes the Ares 5 rocket's second stage (the Earth Depart Stage) as part of the LEO payload. The Soyuz entry vehicle shape is chosen (photographically enlarged by a factor of 7) because this shape has a perfect 35 year track record of successful entries into Earth's atmosphere and because this shape would house an internal centrifuge far better than the conical shape of Apollo modules. Although its entire interior would be protected from the fiery heat of aerocapture, I subdivided that interior space into a crew cabin and another region holding cryogenic LH2/LOX fuel tanks.

The special "hammerhead" launch configuration needed for such a wide payload would be similar to some smaller past launches. It would suffer increased drag while climbing through the lower parts of Earth's atmosphere, but its large size would not impair its subsequent acceleration. With its additional aerodynamic launch losses the LEO mass of this large crew vehicle would need to be somewhat less than 131.8 tonnes, perhaps 120 tonnes. However, the vehicle would be launched empty of any furnishings or provisions. Astronauts following subsequent cargo launches to LEO would assemble its internal centrifuge and also outfit it with other accommodations, including recycling equipment for water and air, and supplies such as food. When completed, the total mass in LEO of this crew vehicle including its reusable heat shield would be around 200 tonnes.

Three additional Ares 5 cargo launches would then provide 395 tonnes of LH_2/LOX fuel which would be transferred in orbit into the crew vehicle's tanks. That would be enough fuel to launch the 200 tonne crew transfer vehicle into a 6-month transfer orbit to Mars. To make this scheme viable, a miniature cryogenic refrigeration system would be needed with the Mars Transfer Vehicle's fuel tanks so that the three Ares vehicles can be launched to LEO separated by several months without losing their cryogenic fuel payloads to boil-off. Such a cryogenic refrigerator may not be too difficult to implement since the insulating vacuum of space will help to minimize its heat load.

If this scheme were followed, the entire transfer vehicle along with its rocket and empty fuel tanks would later be aerocaptured at Mars and left in a parking orbit there until the return trip to Earth. A smaller separate landing vehicle would ferry the crew between the orbiting crew vehicle and the base on Mars' surface where a previously deployed habitation module would await them. It would also ferry up fuel for the return trip.

With a built in *reusable* heat shield similar to the tiles developed for Shuttle instead of an ablative one, the large crew transfer vehicle would be used for multiple aerocaptures. After the return trip from Mars back to Earth the crew transfer vehicle would be aerocaptured into an orbit about Earth instead of landing on Earth's surface. Thus, the crew vehicle would never land and would not need provisions for landing. Instead, an entirely different spacecraft from Earth would rendezvous with it in Earth orbit, pick up its returning crew members and ferry them back to Earth's surface. This would be a far more efficient approach than NASA's present plan to send an unoccupied Orion crew module by rocket all the way to Mars then send it by rocket back to Earth just to achieve

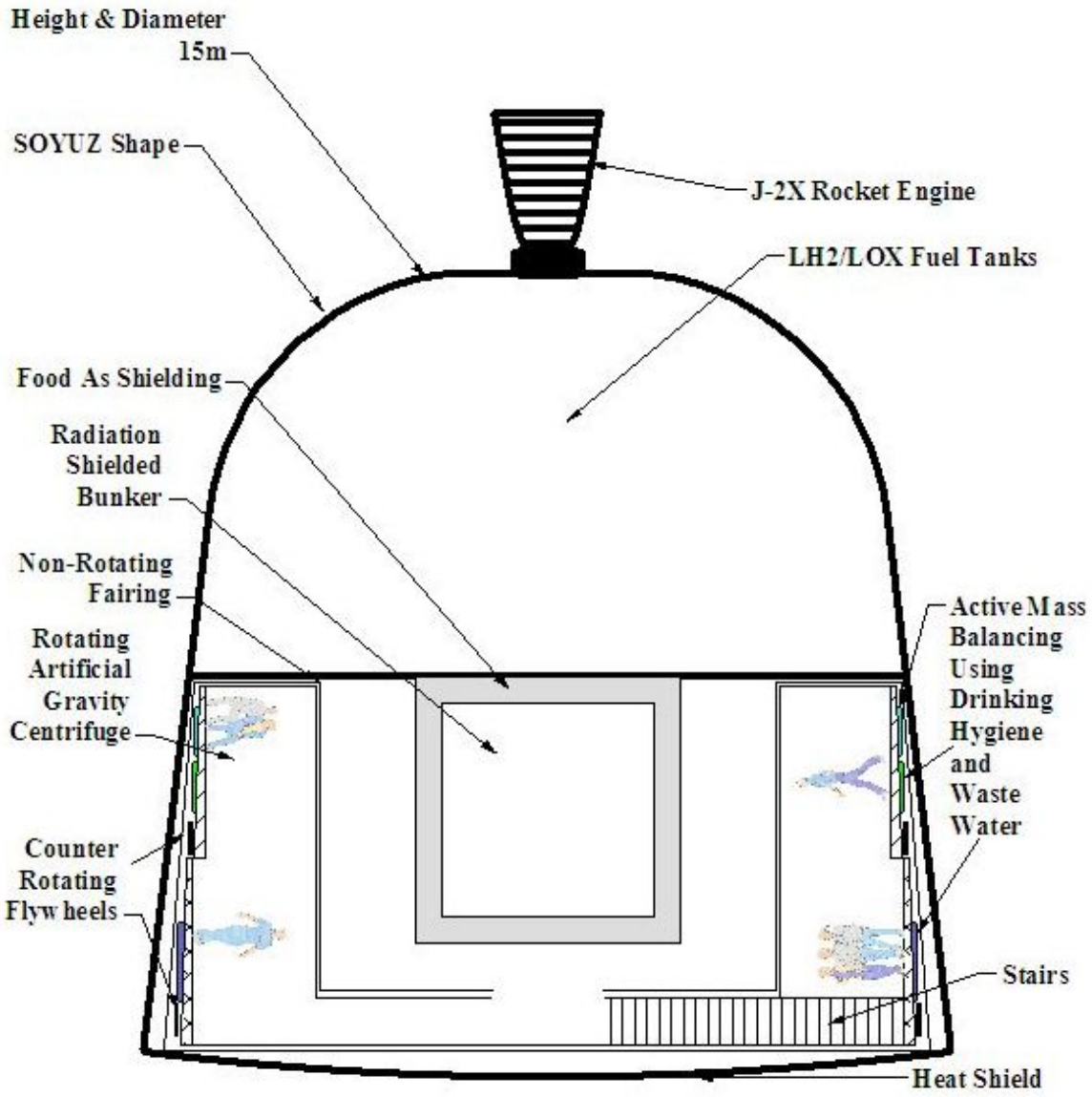


Figure A-24: An Earth/Mars Transfer Vehicle Concept

an Earth landing which could be more efficiently carried out by equipment never traveling to Mars. A further benefit of this scheme is that after its return to Earth the interplanetary crew transfer vehicle with its internal centrifuge and recycling equipment would then be available to restock and reuse on subsequent Mars missions.

A.9 Using In-Situ Martian Resources

Since manned Mars missions will require more mass per crew member than Apollo, will have more crew members per mission than Apollo, and also will require more round-trip rocket Δv per mission than Apollo using similar chemical rocket technologies, it seems plausible they could be more expensive. On the other hand, they may never happen at all if they are much more expensive than Apollo. Therefore, methods to reduce the high costs of manned Mars missions are the central theme which at present underlies planning and will certainly underlie the missions themselves.

To reduce their expense it has been suggested [Zubrin 1996] that rockets should be refueled on Mars via In-Situ Resource Utilization (ISRU), i.e., using martian materials. As the logic of that proposal has been considered, a wide and growing consensus has developed that the use of in-situ resources is the key to affordable manned Mars missions. In ISRU schemes, rocket fuel and oxidizer would be manufactured on Mars for the return trip from Mars to Earth using available martian chemicals. This implies that some type of chemical factory must be imported from Earth along with a power source (e.g., a nuclear reactor) to supply it with its needed energy. There also must be storage tanks to receive rocket fuel and oxidizer as they are manufactured, and cryogenic refrigeration systems needed for storage must be kept operating continuously until the return trip begins.

Prior to the discoveries during the past ten years of abundant water-ice distributed on Mars it was suggested [Zubrin 1996] that hydrogen for the chemical fuel factory could be brought to Mars from Earth. There it would be combined with carbon dioxide from the martian atmosphere available everywhere on Mars' surface, to make methane and free oxygen for rocket engines burning CH_4/LOX. The net of the chemical reactions proposed for this production was as follows.

$$2H_2 + 3CO_2 \rightarrow CH_4 + 2O_2 + 2CO \qquad \text{A-17}$$

As implied by Eq. A-17, each 4 tonnes of hydrogen from Earth would be combined with 132 tonnes of carbon dioxide from Mars' air and energy to yield 80 tonnes of rocket fuel, including 16 tonnes of liquid methane and 64 tonnes of LOX, while the 56 tonnes of carbon monoxide waste product would be discharged back into the air. Although the chemistry of this scheme is correct (as was demonstrated by a small prototype), I am skeptical about its practicality. It would be nontrivial to keep liquid hydrogen at cryogenic temperatures from Earth departure through the transfer orbit, then through a fiery entry into Mars' atmosphere and finally through a long period of storage on Mars'

surface until it is all used. Refrigerating cryogenic fluids in space may be the easiest of these since a vacuum is a good thermal insulator. Thermal insulation would dominate issues in aerodynamic entry and surface storage of cryogenic liquid hydrogen.

Fortunately, it now appears feasible to rely on abundant water resources to obtain the hydrogen needed since water exists in some locations on Mars' surface. This simplifies the endeavor but also elevates the importance of locations with abundant water resources. It would be prudent to remotely deploy and operate the chemical factory and its nuclear power source in advance of manned missions so that it can entirely fill its storage capacity with manufactured fuel and oxidizer for the return trip home before travelers even depart from Earth enroute to Mars. Subsequent missions could repeat this pattern so the nuclear reactor and chemical fuel factory could operate for many missions.

There is not yet a clear consensus concerning the best choice for the chemical rocket fuel combination to manufacture on Mars. The main controversy is whether to use LH_2/LOX or CH_4/LOX. Although abundant water resources now appear to exist on Mars, Zubrin continues to advocate his previously proposed scheme to bring liquid hydrogen from Earth and combine it with martian air to manufacture CH_4/LOX fuel. He correctly points out that by using his scheme, rocket fuel for the return trip could be made at any landing site without regard to its proximity to water-ice deposits. Zubrin is not alone in advocating the use of this fuel. Indeed, many technical people are convinced that CH_4/LOX is the best rocket fuel for Mars.

If there were no nuclear engines providing the capability for long range (i.e., planetary) surface mobility, then Zubrin's scheme would be needed in order to visit and scientifically study different regions. However, if the nuclear engines of the present work are feasible and can provide long range surface mobility then this argument for the use of CH_4/LOX fuel becomes much less compelling.

It might even be feasible to manufacture hydrazine and nitrogen tetroxide on Mars from entirely martian resources. Since the martian atmosphere is 2.7% nitrogen gas, unlimited quantities of pure nitrogen could be obtained by cryogenic separation of the martian air (driven by an electric energy source). The nitrogen would then be combined with martian water in the *net* chemical reactions of Eq. A-19. Each 140 tonnes of atmospheric nitrogen combined with 72 tons of water would yield 212 tonnes of rocket fuel, as 120 tonnes of hydrazine and 92 tonnes of nitrogen tetroxide. The actual chemical manufacturing process may be complicated and the resulting rocket fuel of lower performance, but at least this rocket fuel combination would not need active refrigeration.

$$5N_2 + 4H_2O + \rightarrow 4N_2H_2 + N_2O_4 \qquad \text{(A-19)}$$

However, the exhaust velocity using this fuel is low enough that it has no vocal advocates at present for Mars missions.

In comparing LH_2/LOX with CH_4/LOX fuels it is useful to consider their properties as cryogenic liquids. Table A-6 compares their boiling point temperatures at Earth's normal sea-level pressure and also their mass densities at those boiling points.

The chemical reactions taking place in the two alternative rockets are as follows:

$$CH_4 + 2O_2 \rightarrow 2H_2O + CO_2 \quad \text{(A-18)}$$

$$2H_2 + O_2 \rightarrow 2H_2O \quad \text{(A-19)}$$

Using these and the densities in Table A-5, the calculated necessary cryogen *volumes* are summarized in Table A-7, which shows that CH_4/LOX tanks would be only half the volume of LH_2/LOX tanks for the same propellant mass. Table A-5 suggests that cryogenic refrigeration for CH_4/LOX might also be easier to provide than refrigeration for LH_2/LOX because hydrogen's liquid boiling point temperature is considerably lower than oxygen's whereas the liquid boiling point temperature of CH_4 (methane) is near to, but slightly above, oxygen's. Although a somewhat larger mass of CH_4/LOX than LH_2/LOX would be needed to provide the same rocket Δv, this comparison of fluid properties indicates an advantage in handling and storage for CH_4/LOX over LH_2/LOX **.**

Using martian water, it would be simple to produce hydrogen and oxygen by electrolysis. The apparatus needed to electrolyze water is simple, compact, and rugged. Each tonne of water ice would be converted into one tonne of high performance LH_2/LOX fuel. Alternatively, martian water could be combined with martian carbon dioxide and energy to produce methane and oxygen, following the chemical reaction of Eq. A-20:

$$2H_2O + CO_2 \rightarrow CH_4 + 2O_2 \quad \text{(A-20)}$$

Here, 36 tonnes of martian water are combined with 44 tonnes of martian carbon dioxide and energy to yield 80 tonnes of rocket fuel, as 16 tonnes of methane and 64 tonnes of oxygen. Without relying on biological processes the chemical processes needed to industrially accomplish the net reaction of Eq. A-20 are more complicated than electrolysis alone, and their necessary apparati are larger and more sophisticated. However, there is no doubt they can be accomplished.

NASA considered adopting CH_4/LOX rocket fuel for their planned return to Luna and rejected it because switching from LH_2/LOX to CH_4/LOX would require more engineering development but would *reduce* rocket performance. At the present time, NASA's plans for rockets to use in return legs of round-trip Mars missions are not firm. NASA's recent public statements have focused on the proposed use of CH_4/LOX rockets.

My own view is that LH_2/LOX should be the rocket fuel manufactured on Mars. Since abundant water has been found on Mars, it is a better approach. It is far easier to separate water into hydrogen and oxygen by electrolysis than any of the other proposed chemical rocket fuel manufacturing schemes, e.g., for methane, so fuel manufacturing equipment

Table A-6: Properties of Cryogens In Rocket Fuels Proposed For Mars Returns

Cryogenic Liquid	**Normal Boiling Point (Kelvins)**	**Liquid Density At Normal Boiling Point Temperature (g/cm^3)**
H_2	**20.28**	**0.0708**
O_2	**90.188**	**1.141**
CH_4	**111.668**	**0.4224**

Table A-7: Tank Sizes Per Tonne Propellant For Different Fuels

	LH_2/LOX (m^3/tonne)	**CH_4/LOX (m^3/tonne)**
H_2	**1.569**	**---**
O_2	**0.779**	**0.7008**
CH_4	**---**	**0.4734**
Totals	**2.348**	**1.1742**

would be simplified. The resulting LH_2/LOX would provide the highest exhaust speed of all practical chemical rocket fuels. In addition, LH_2/LOX has a large and growing experience base as a rocket fuel, whereas at the present time CH_4/LOX has none.

Most importantly, since LH_2/LOX will be used by the Ares 5 rocket's Earth Depart J-2X rocket stage (or by my proposed crew vehicle concept) launching the mission into a transfer orbit to Mars, that same J-2X rocket engine along with its empty LH_2/LOX fuel tanks will arrive intact at Mars. If it is aerocaptured then LH_2/LOX fuel production on Mars can power that same J-2X rocket engine for the trip back to Earth. Since the J-2X has been optimized for LH_2/LOX, it seems unlikely that CH_4/LOX would work as well in the same engine.

A key link needed to complete my proposed scheme is a reusable single-stage-to-orbit (SSTO) rocket vehicle able to robotically ferry either several crew members or a mass of LH_2/LOX fuel between the base on Mars' surface and the crew vehicle in orbit about Mars. By using a SSTO there would be no discarded rocket stages or other expendable parts. After refilling the SSTO's fuel tanks at the Mars base it would be ready to fly again without additional refurbishing or maintenance. Such reusability requires that parachutes be avoided in its design by slightly increasing rocket propulsion's role in landing.

Although present technology has not yet succeeded in designing a reusable SSTO for travel between Earth's surface and low Earth orbit, it would be far easier to design such a reusable SSTO for the much smaller rocket Δv required for travel between Mars' surface and low Mars orbit. As indicated in Table A-2, the Δv requirements to reach low Mars orbit are only about 3.5 km/sec plus small launch losses, less than half of the approximately 9 km/sec including launch losses needed to reach low Earth orbit. Even with some additional rocket Δv requirements allocated for the final seconds of landing, the use of LH_2/LOX with its high exhaust speed should allow a small but fully reusable SSTO to be deployed without requiring use of any expendable equipment.

Eq. A12 predicts that a single-stage rocket carrying just 5 times as much fuel as its empty mass can accelerate itself and an additional payload mass equal to its empty mass through a Δv equal to 125 % of its rocket exhaust speed. Thus, using LH_2/LOX fuel with a 4.3 km/sec exhaust speed would provide an SSTO with total Δv capability of (4.3)(1.25)=5.39 km/sec. This seems ample to carry the payload in a climb up to 200 km altitude, accelerate it to the 3.47 km/sec speed of the orbiting crew vehicle, exchange the ascending payload with an equally massive descending payload if there is one for the return trip down, and still have over 1.5 km/sec Δv capability remaining for a propulsive rocket landing after a hypersonic reentry. Since the SSTO's entry speeds would be slow compared to interplanetary arrival speeds, its reusable heat shield could be minimal.

Depending on transfer orbit details, between 120 and 200 tonnes of LH_2/LOX fuel would be needed in low Mars orbit to send a 200 tonne crew transfer vehicle back to Earth. With an empty SSTO mass as little as 2 tonnes (the mass of a car) ferrying daily LH_2/LOX fuel payloads of 2 tonnes each from Mars surface to the orbiting crew vehicle's

fuel tanks (while consuming 10 tonnes of fuel per round-trip), all fuel needed for the return trip to Earth could be in orbit after no more than 100 SSTO flights. This would easily fit into the 500 day duration of a crew's surface stay on Mars. A two tonne payload capability for the SSTO would also be adequate to ferry crew members between the crew vehicle and the ground in small groups over several days.

A.9.1 Implications of Using In-Situ Martian Resources

Adoption of an in-situ rocket fuel production strategy implies deploying on the martian surface a chemical factory and an associated nuclear reactor to operate it. This has a profound effect on the context assumptions appropriate to take as "boundary conditions" for evaluating future Mars surface vehicle engine requirements. Since a rocket fuel chemical factory and its nuclear reactor are significant deployments that could support many subsequent round-trip missions to Mars, the deployment site will almost certainly become a permanent Mars base at which subsequent missions will need to land with pinpoint spatial precision. Whether the choice is to produce the presently preferred LOX/methane rocket fuel or the more energetic but colder and less compact LOX/hydrogen rocket fuel, the Mars surface site chosen for this base must be close enough to a source of water-ice so that tele-operated robots can harvest it. These robots will need to have engines sufficiently powerful for their water-ice mining role, and in addition to mechanical strength may require the use of portable heat to harvest water. Thus, mining is a natural role for the nuclear engines which are the subject of this thesis.

The base's stationary nuclear reactor will provide crews with welcome heat in the cold Martian environment, and any excess electricity from it will also be available for their use. By increasing the production capacity of the chemical factory which will continuously produce both oxygen and either methane or hydrogen, some of its product can be diverted to other uses. For instance, small light-weight spark-ignition internal combustion engines with dual fuel/oxygen tanks could be used with either methane or hydrogen fuel to power either all-terrain vehicles or small power tools, as long as they operate sufficiently close to the base for refueling. If hydrogen is the fuel produced then more efficient (but heavier) hydrogen/oxygen fuel cells could be used to directly produce portable electricity while retaining the resulting water for reuse. Some of the produced oxygen could be used for breathing. The robots harvesting water ice for fuel production could also harvest additional water for crew members to use for drinking and washing.

As in Apollo vehicles, the breathing air mixture for Mars missions will be almost pure oxygen at a low pressure chosen to match Earth's oxygen partial pressure near sea level, likely near 22 kPa (i.e., 3.2 psi). Such standard use of low pressure oxygen simplifies necessary pressure boundaries, from deep space vehicles to astronauts' pressure suits to enclosed mobile surface vehicles to buildings at the base.

Although pressure-tight habitation modules constructed on Earth will have been delivered to the base on Mars before the first crew's arrival there, they will need to be augmented. Additional pressurized spaces will be needed to conduct "shirt-sleeve" science operations

at the base, which will include both tele-operation of robots and laboratory analysis of samples. Extremely large pressurized spaces will be constructed at the base by excavation, the simplest possible building scheme for Mars.

The simplest building scheme envisioned for Mars combines an ultra-light-weight inflatable structure with ground excavation. Construction starts by excavating a very large brick-shaped room in the martian ground to a depth of about three meters (i.e., about 10 feet). A close-fitting light-weight brick-shaped liner balloon brought from Earth, air tight but not sufficiently massive to hold 22 kPa internal air pressure based on its skin tension alone, is inserted into that hole. At this point, the ground fully supports five faces of the brick-shaped liner balloon by mechanically pressing back on its floor and walls, but the balloon's ceiling remains unsupported. Loose soil is then evenly piled on top of the balloon while the balloon simultaneously is being inflated with oxygen gas. The internal gas pressure is increased in step with the increasing weight of "top soil" in order to maintain the liner balloon's brick-like shape without excessive skin tension. In the low martian gravity, the depth of 2000 kg/m^3 soil needed to balance 22 kPa internal oxygen pressure is about 3 meters (i.e., about 10 ft), so the soil needed to pile on top of the liner balloon is the same soil that was excavated earlier. When finished, the entire room will be supported by its own internal air pressure which will slightly exceed the weight of "topsoil" piled on top, and by shear stresses in adjacent martian soil. The role of the liner balloon, which will have a low level of mechanical tension, will be to seal the room to avoid air leakage. Since the room's structural strength in this scheme depends on maintaining internal air pressure to slightly more than balance the weight of "topsoil", the balloon must be connected to a reliable control system regulating internal air pressure in addition to the standard life support functions of removing carbon dioxide, providing fresh oxygen, regulating temperature and controlling humidity. Provisions for the room's external door/airlock would need to have been built into the imported liner balloon and integrated with the overall excavation.

A windfall benefit of using excavation to constuct pressure-tight spaces for human occupancy is reduced radiation exposure. Radiation from galactic or solar sources in such spaces will have been attenuated by passage through the 3 meter thick topsoil and may as a result be near the background radiation levels on Earth's surface.

More complex building schemes might perhaps be feasible to avoid needing any liner balloon brought from Earth. For example, a room could again be excavated in the martian soil but then covered by a vaulted ceiling of precisely cut martian rock (using fundamentally medieval technology), thus forming a strong but not airtight structure. A plasma torch could then be operated to spray molten soil material against the room's walls, ceiling, and floor in order to seal all cracks. The room's external door/airlock would still need to be imported from Earth and installed and sealed in some way.

Although very large pressurized spaces on Mars could be constructed using little construction material mass brought from Earth, such construction would require excavation of many tonnes of martian soil. Powerful engines would be needed to power

the backhoes and bulldozers for this work. For instance, construction of a moderately sized pressurized room with just 100 square meters (1,076 ft^2) of floor area would require first excavating about 600 tonnes of soil, then piling that 600 tonnes back onto the imported liner balloon's roof. These are roles for mobile nuclear powered engines.

Eventually, more sophisticated building materials could be made via industrial production in which nuclear-generated electricity would be used along with equipment at the base to process martian mineral resources. Bricks could be made by sufficient electrical heating of moistened soil. If sufficiently pure silicates were found nearby, transparent glass could be made by electrically melting them. Also, some of the hydrogen gas produced by electrolysis of harvested water could be combined with extremely heated soil rich in iron-oxide to remove the oxygen, yielding metallic iron.

A.10 Scientific Activities

The main research activities for scientists on Mars will focus on obtaining and analyzing samples of the martian regolith (i.e., the ground) Some samples such as rocks and soil will come from surface locations while others will be obtained using deep drilling equipment. The samples will be taken from numerous geographical locations all over the planet. Since the equatorial radius of Mars is 3,380 km, most samples will be collected at locations quite remote from the base, e.g., thousands of kilometers distant. Many of the samples will be obtained using mobile robotic devices tele-operated from the Mars base by the scientists, since as few as four orbiting relay satellites can provide high bandwidth communications with negligible round-trip time delay between any two points on the martian globe. Other samples may be obtained by scientists physically traveling to the sample locations in suitable long range vehicles. Although in principle these martian regolith samples could be sent back to Earth for analysis in Earth's laboratories, the sheer quantity of sample material which will be obtained will preclude sending most of the samples back to Earth. Instead, sophisticated laboratory equipment, e.g., optical, electron, and STM microscopes, will be deployed at the Mars base and used by scientists there in pressurized shirt-sleeve room environments to analyze the samples.

Scientists at the Mars base may also tele-operate lightweight unmanned aircraft for basic research in order to make visual observations and atmospheric measurements not possible from either orbit or ground. For examples, an aircraft could be flown through the deep Valles Marinaris canyon while examining its walls, or alternatively could be flown through the mile-deep gash which cuts through the northern polar icecap.

Some types of data collection such as for seismology or for atmosphere studies will be accomplished by deploying remote sensors to relay measurements to orbiting satellites reporting directly to Earth. Crew members will deploy such sensors directly or by using tele-operated robots, but analyses of their data will most efficiently proceed on Earth.

In addition to their basic Mars research, each expedition's crew members will spend some portion of their time engaged in applied research invested in improving the effectiveness of later expeditions. One major theme will focus on how to best grow green plants at the base so that locally produced food and breathing oxygen can displace imports from Earth on later missions. Other themes will focus on how to best process martian mineral resources in order to locally fabricate needed materials and products. There also will be interest in finding effective ways to expand chemical storage capacity for important low pressure liquids such as liquid water, liquid oxygen, liquid methane, and/or liquid hydrogen, even though their natural liquid temperatures are different from that of the martian surface environment.

APPENDIX B
B MATLAB CALCS: EARTH/MARS ORBIT TRANSFER

This appendix is configured as a MATLAB/MSWORD Notebook in order to internally implement computations to investigate characteristics of transfer trajectories between Earth and Mars. Formatting within this appendix follows MATLAB/MSWORD Notebook norms since they contain specially formatted "cells" of MATLAB code written in a particular special font and delineated by special hidden characters. MATLAB/MSWORD Notebook documents also contain output results from the specified computations, which may be graphical plots or numerical tables using a different special font. Such documents deviate from conventions for easily readable material since they are essentially executable computer codes and their resulting outputs along with some interspersed explanatory material. For this thesis, they allow the reader to perform a detailed audit of calculations made and they also make it possible for the reader to redo the calculations by use of the "cut and paste" function.

The purpose of orbit transfer investigations for the present nuclear engine design project is to find out directly what are the constraints imposed by the realities of Earth/Mars transportation. This direct investigation seems a superior alternative to relying on published statements of so-called opinion leaders, since those statements frequently fail to coalesce into a consensus. In any event, these calculations are not difficult to carry out. The differential equations for orbital transfer were solved in closed form centuries ago and subtle numerical improvements to the resulting formulae have evolved throughout the 1800s and 1900s. Thus, the orbit calculations in this appendix are largely the numerical evaluation of formulae extracted from relevant textbooks. Since each of the plots in this appendix was produced directly by executing the included MATLAB code that precedes it, none have been assigned figure numbers herein. However, each of the plots copied and pasted into locations outside this appendix has been assigned a figure number there.

The starting point is a listing the orbital elements of the two planets along with other data needed for these computations, as quoted by alternate sources. These are collectedhere in Table B-1. [Bate, Mueller and White 1971, 360-361], [Duffett-Smith 1988, 105]]

It should be noted that two of these quoted quantities, the longitude at epoch and the longitude at periapsis (i.e., perihelion), are defined in a traditional but strange way, which has the merit that these quantities remain well defined even if orbital and ecliptic planes coincide. Although each quantity has angle measure, each is the sum of separate angles measured in what are usually different planes, i.e., one in the ecliptic plane and the other in the orbital plane. Thus, there is no geometric angle matching either of them. The longitude of perihelion is defined as the sum of the longitude of the ascending node (which is measured in the ecliptic plane) plus the angle from the ascending node to the perihelion (measured in the orbital plane in the direction of motion). The longitude at epoch is the sum of that and the true anomaly angle at the epoch time, measured in the orbital plane in the direction of motion from the perihelion to the orbiting object.

Table B-1: Orbital Elements and Other Astronomical Data for Earth and Mars

Planet	**Earth**		**Mars**	
Data Source	**Bate et al**	**[Duffett-Smith]**	**[Bate et al]**	**[Duffett-Smith]**
Epoch Date	**1969 June 28.0**	**1990 January 0.0**	**1969 June 28.0**	**1990 January 0.0**
Longitude at Epoch Date	**276.117°**	**99.403308°**	**265.096°**	**240.739474°**
Orbital Period (tropical years)**	**1.000**	**1.00004**	**1.881**	**1.880932**
Orbital Inclination to Ecliptic Plane	**0.000°**	**0°**	**1.850°**	**1.849736°**
Semi-major axis of the orbit (A.U.)*	**1.000**	**1.00000**	**1.524**	**1.523688**
Eccentricity of the orbit	**0.0167**	**0.016713**	**0.0934**	**0.093396**
Longitude at Perihelion	**102.416°**	**102.768413°**	**335.497°**	**335.874939°**
Longitude of the ascending node	**undefined**	**undefined**	**49.322°**	**49.480308°**
Equatorial radius (km)	**6378**		**3380**	
Planetary Gravitational Source, □ (km^3/sec^2)**	**$3.986 * 10^5$**		**$4.305 * 10^4$**	

***1 A.U.=$1.4959965 * 10^8$ kilometers**

**** 1 tropical year = 365.242191 mean solar days**

Before using these tabulated data it is useful to review the formulae for orbital computations. The classical orbital elements are as follows:

$a \equiv$ semi - major axis
$e \equiv$ eccentricity
i $\equiv$ inclination
$\Omega \equiv$ longtitude of ascending node
$\omega \equiv$ argument of periapsis
$T \equiv$ time of periapsis passage

Additionally, textbooks on orbital transfer define some related and derived parameters.

$\mu \equiv$ central gravitation parameter (proportional to central mass)
$p \equiv a\left(1-e^2\right) \equiv$ semi - latus rectum $\equiv$ parameter of the orbit
$b \equiv$ semi - minor axis (if ellipse)
$c \equiv$ half of inter - focus distance (if ellipse)

Note that

$$p = a\left(1-e^2\right) \tag{B-1}$$

$$e = \frac{c}{a} \tag{B-2}$$

$$a^2 = b^2 + c^2 \tag{B-3a}$$

which imply that for ellipses,

$$b = a\sqrt{1-e^2}$$
$$c = ae \tag{B-3b,c}$$

The polar equation of conic section "Keplerian" orbit is:

$$r(t) = \frac{p}{1+e\cos\phi(t)} \tag{B-4}$$

where

$r(t) =$ distance of object from central focus, a function of time, t.
$\phi(t) =$ "true anomaly", the central angle from periapses to object, a function of time, t

Kepler's formulation involves the definition of two other derived "angles", which must be expressed in radian measure (not degrees), and which change as a function of time:

$$\cos E(t) = \frac{e + \cos\phi(t)}{1 + e\cos\phi(t)} \tag{B-5}$$

$$M(t) = E(t) - e\sin E(t) \quad \text{(Kepler's equation)} \tag{B-6}$$

where
$E(t)$ = "Eccentric anomaly" angle, a function of time, t
$M(t)$ = "Mean anomaly" angle, a function of time, t

and where the connection with time is given by

$$M(t) = n*(t - T) \tag{B-7}$$

where

$$n = \sqrt{\frac{\mu}{a^3}} \tag{B-8}$$

Given a time, t, and the classical orbital elements, we first calculate the Mean Anomaly via Equations (B-8) and (B-7). We then calculate the Eccentric Anomaly via Equation (B-6), which is a transcendental equation that must be solved numerically by iterative methods. We then calculate the true anomaly by inverting Equation (B-5). We note that in order to invert Equation (B-5) we need another relation linking eccentric and true anomaly angles. That is provided by the following:

$$\sin\phi(t) = \frac{\sqrt{1-e^2}\,\sin E(t)}{1 - e\cos E(t)} \tag{B-9}$$

Then we calculate the distance by evaluating Equation (B-4).

Then we calculate the position in the heliocentric-ecliptic coordinate system, denoted as XYZ. They are found through successive coordinate system rotations.

For this development, we fix the origins of all coordinate systems at the central gravitational focus. Start with an X'''Y'''Z''' coordinate system whose X''' axis is aligned with the radius vector and Z''' axis aligned with the angular momentum vector, which results in the X'''Y''' plane being the orbital plane. Then we define a second coordinate system, X''Y''Z'', by rotating about the Z''' axis through an angle of $-(\omega + \phi(t))$. This

leaves one axis unchanged, i.e., Z'"=Z". It also repositions the angular position of the radius vector within the X"Y" plane to be the sum of the true anomaly and the argument of perigee, which we note is the object's rotation angle from the ascending node. We then define a third coordinate system, X'Y'Z', by rotating about the X" axis by the inclination angle, *i*. This brings the Z' axis perpendicular to the ecliptic plane. Finally, we obtain the heliocentric-ecliptic coordinate system, XYZ, by rotating about the Z' axis through the angle, -Ω.

The rotations are written in matrix form as follows:

$$\begin{bmatrix} X(t) \\ Y(t) \\ Z(t) \end{bmatrix} = \begin{bmatrix} \cos\Omega & -\sin\Omega & 0 \\ \sin\Omega & \cos\Omega & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos i & -\sin i \\ 0 & \sin i & \cos i \end{bmatrix} * \begin{bmatrix} \cos(\omega+\phi(t)) & -\sin(\omega+\phi(t)) & 0 \\ \sin(\omega+\phi(t)) & \cos(\omega+\phi(t)) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r(t) \\ 0 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\Omega & -\sin\Omega & 0 \\ \sin\Omega & \cos\Omega & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos i & -\sin i \\ 0 & \sin i & \cos i \end{bmatrix} * \begin{bmatrix} \cos(\omega+\phi(t)) \\ \sin(\omega+\phi(t)) \\ 0 \end{bmatrix} \qquad \text{(B-10)}$$

$$== \begin{bmatrix} \cos\Omega & -\sin\Omega\cos i & \sin\Omega\sin i \\ \sin\Omega & \cos\Omega\cos i & -\cos\Omega\sin i \\ 0 & \sin i & \cos i \end{bmatrix} * \begin{bmatrix} \cos(\omega+\phi(t)) \\ \sin(\omega+\phi(t)) \\ 0 \end{bmatrix}$$

or

$$\omega \begin{bmatrix} X(t) \\ Y(t) \\ Z(t) \end{bmatrix} = \begin{bmatrix} \cos\Omega\cos(\omega+\phi(t)) - \sin\Omega\cos i\sin(\omega+\phi(t)) \\ \sin\Omega\cos(\omega+\phi(t)) + \cos\Omega\cos i\sin(\omega+\phi(t)) \\ \sin i\sin(\omega+\phi(t)) \end{bmatrix} r(t) \qquad \text{(B-11)}$$

Thus, given any time, t, and the orbital elements listed in Table 1, we can calculate the positions in heliocentric-ecliptic coordinates of Earth and/or Mars. We will start our calculations with the specified required times to leave one planet and to arrive at the other planet, and we shall use the above procedures to determine the locations of the two planets at the respective times.

We also need to calculate planetary velocities at the same times. The velocity vector is in the orbital plane. It has one component, $\dot{r} \equiv \frac{dr}{dt}$, directed from the focus through the

object, and another perpendicular component, $r\dot{\nu} \equiv r\frac{d\nu}{dt}$. We can do this using some relations involving the specific angular momentum. We have

$$h = r^2\dot{\nu} = \sqrt{\mu p} = \sqrt{\mu a\left(1-e^2\right)} \tag{B-12}$$

Differentiating Equation (4) with respect to time, t, we get:

$$\dot{r}(t) = \frac{p\sin\phi(t)}{\left(1+e\cos\phi(t)\right)^2}\dot{\phi}(t) \tag{B-13}$$

Substituting (12) into (13), we get:

$$\dot{r}(t) = \frac{p\sin\phi(t)}{\left(1+e\cos\phi(t)\right)^2}\frac{h}{\left(r(t)\right)^2} = \frac{hp\sin\phi(t)}{\left(1+e\cos\phi(t)\right)^2}\left(\frac{1+e\cos\phi(t)}{p}\right)^2 = \frac{h}{p}\sin\phi(t) \tag{B-14}$$

Combining Eq. (4) with (12) gives

$$r(t)\dot{\nu}(t) = \frac{h}{r(t)} = \sqrt{\mu p}\,\frac{1+e\cos\phi(t)}{p} = \sqrt{\frac{\mu}{p}}\left(1+e\cos\phi(t)\right) \tag{B-15}$$

Equations (13) and (14) give formulae for calculating the velocity components in the X''' and Y''' directions, respectively. The rotation matrices of Equation (9) can then be used to transform them into heliocentric-ecliptic coordinates.

According to [Bate. Mueller and White 1971, 190], a computational difficulty occurs with near-parabolic orbits for which the eccentricity approaches 1 while the semi-major axis approaches infinity. For such cases, calculation of p by evaluation of Equation (1) involves multiplying near-infinite numbers by near-zero numbers, so for such cases equations (14) and (15) become inaccurate. The authors of this reference, in addition to other sources, derive some more universal equations which avoid that difficulty. Also, for hyperbolic orbits, Kepler's equation, Eq. (6), must be replaced by another transcendental relation, one involving the sinh function instead of the sine function. The universal variables formulation avoids that difficulty as well, and provides a single formulation covering all possible central force gravitational orbits, elliptical, parabolic, or hyperbolic. The universal variables formulation seems superior for the problems of computing the transfer trajectories, and is therefore implemented for that purpose later below.

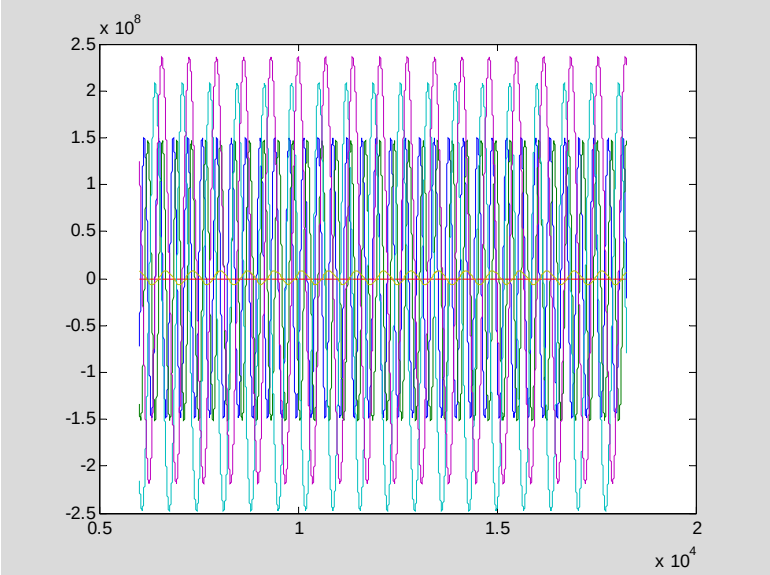
On the other hand, the Keplerian formulation is just fine for the purpose of calculating the positions and velocities of Earth and Mars at any specified times, since their orbits are ellipses of low eccentricity. That is done in the following MATLAB scripts.

```
clear;
daysperyear=365.242191;
kmperAU=149597870.;
Tp_yrs=[1.00004  1.880932];
incl_deg=[0.  1.849736];
a_AU=[1.00000  1.523688];
eccentricity=[0.016713  0.093396];
longt_asc_node_deg=[0  49.480308];
longt_peri_deg=[102.768413  335.874939];
longt_epoch_deg=[99.403308  240.739474];
t_epoch=datenum('00-Jan-1990');
lan=longt_asc_node_deg*pi/180.;
trueanomaly_epoch=[longt_epoch_deg-longt_peri_deg]*pi/180.;
E_epoch=acos((eccentricity+cos(trueanomaly_epoch))./(1+eccentricity.*co
s(trueanomaly_epoch)));
M0=eccentricity.*sin(E_epoch)-E_epoch;
p=kmperAU*a_AU.*(1-eccentricity.^2);
arg_peri=(longt_peri_deg-longt_asc_node_deg)*pi/180.;
carg_peri=cos(arg_peri);sarg_peri=sin(arg_peri);
lan=longt_asc_node_deg*pi/180.;
clan=cos(lan);slan=sin(lan);
incl=incl_deg*pi/180.;
cincl=cos(incl);sincl=sin(incl);
```

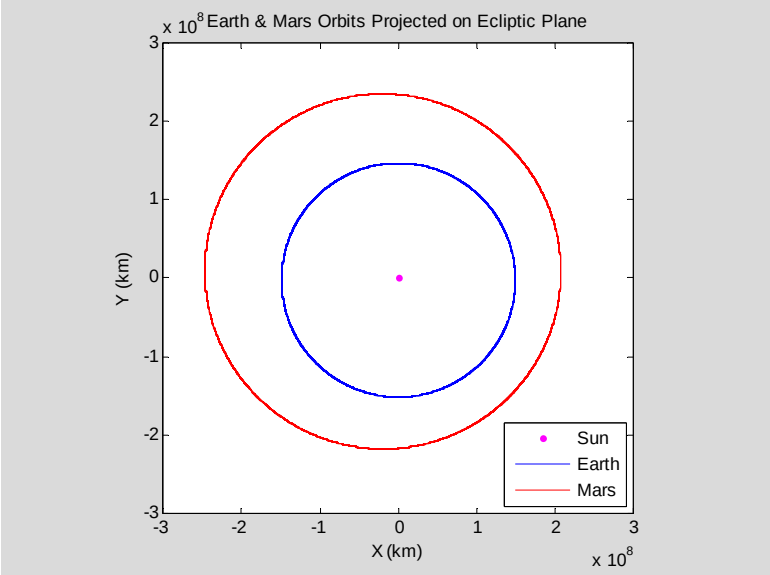
Now that the orbit elements data has been entered and/or calculated, we proceed to calculate planetary locations at a specified list of times, t.

```
format long g,
tfirst=datenum('23-May-2006')-t_epoch;
tlast=datenum('31-Dec-2039')-t_epoch;
t=linspace(tfirst,tlast,10000)';
ecc=ones(size(t))*eccentricity;
M=ones(size(t))*M0+2*pi*t/daysperyear*(1./Tp_yrs);
E=M;
for i=1:10;
 delta=E-ecc.*sin(E)-M;
  if abs(max(max(delta)'))<10*eps
   break
  end
 E=E-delta./(1-ecc.*cos(E));
end
i,abs(max(max(delta)')),
phi=atan2(sqrt(1-ecc.^2).*sin(E)./(1-ecc.*cos(E)),(cos(E)-ecc)./(1-
ecc.*cos(E)));
r=ones(size(t))*p./(1+ecc.*cos(phi));
inplanecoords=[r(:,1).*cos(arg_peri(1)+phi(:,1))
r(:,1).*sin(arg_peri(1)+phi(:,1))  zeros(size(t))
r(:,2).*cos(arg_peri(2)+phi(:,2))  r(:,2).*sin(arg_peri(2)+phi(:,2))
zeros(size(t))];ROT=[cos(lan(1)) sin(lan(1)) 0 0 0 0;
-sin(lan(1))*cos(incl(1)) cos(lan(1))*cos(incl(1)) sin(incl(1)) 0 0 0;
sin(lan(1))*sin(incl(1))  -cos(lan(1))*sin(incl(1)) cos(incl(1)) 0 0 0;
0 0 0 cos(lan(2)) sin(lan(2)) 0; 0 0 0 -sin(lan(2))*cos(incl(2))
```

```
cos(lan(2))*cos(incl(2)) sin(incl(2)); 0 0 0 sin(lan(2))*sin(incl(2))
-cos(lan(2))*sin(incl(2)) cos(incl(2))];XYZeXYZm=inplanecoords*ROT;

i =
    10
ans =
     5.6843418860808e-014

plot(t,XYZeXYZm)
```

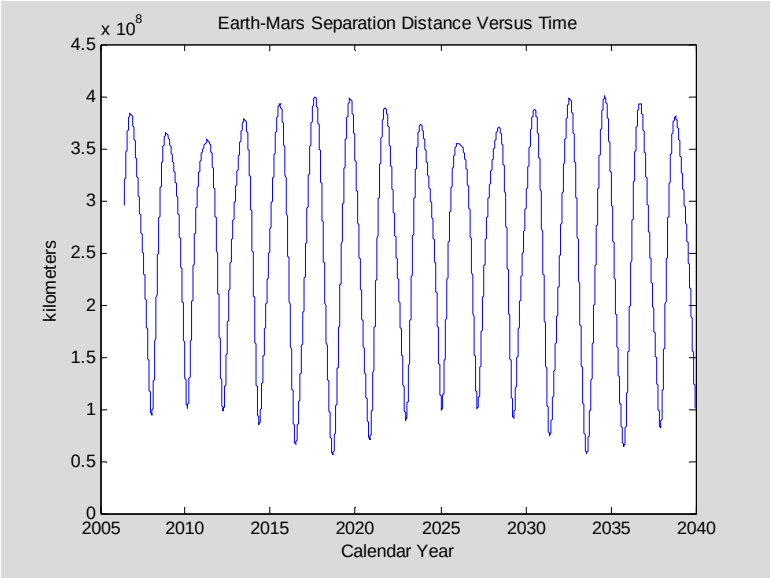



```
plot(0,0,'m.',XYZeXYZm(:,1),XYZeXYZm(:,2),'b',XYZeXYZm(:,4),XYZeXYZm(:,
5),'r'),xlim([-3e8 3e8]),ylim([-3e8 3e8]),xlabel('X (km)'),ylabel('Y
(km)'),title('Earth & Mars Orbits Projected on Ecliptic
Plane'),legend('Sun','Earth','Mars','Location','SouthEast'),axis square
```

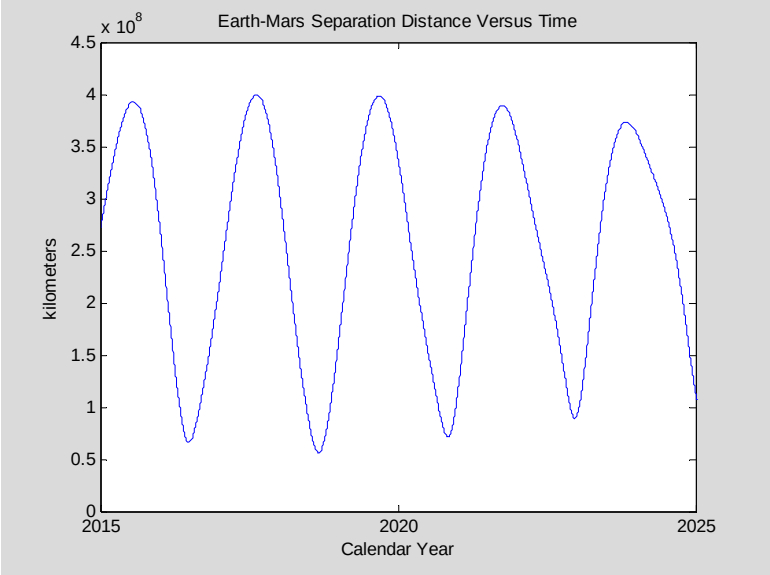


Using the calculated position histories, the distance history can be directly calculated.

```
D_e_m=sqrt((XYZeXYZm(:,4)-XYZeXYZm(:,1)).^2+(XYZeXYZm(:,5)-
XYZeXYZm(:,2)).^2+(XYZeXYZm(:,6)-XYZeXYZm(:,3)).^2);
plot(t+t_epoch,D_e_m),ylim([0 4.5e8]),datetick('x'),xlabel('Calendar
Year'),ylabel('kilometers'),title('Earth-Mars Separation Distance
Versus Time')
```




```
xlim([datenum(2015,0,0)  datenum(2025,1,1)])
```



Note that the cartesian coordinate orbital positions can alternatively be calculated using the PQ vector formulation, which conceptually avoids the use of coordinate rotations. The periapsis vector, P, and the perpendicular to it in the orbital plane, Q, are first calculated in inertial coordinates. Then the orbital position and velocity are calculated by evaluating simple functions of the eccentric anomaly angle. These functions multiply by the two vectors and the products are summed. Although the actual calculations may be of about the same complexxity, this following approach seems more easily understood, and therefore, more easily debugged. Furthermore, references give formulas for the velocity using this method, and the velocities are needed.

We start by calculating three unit vectors, c_hat is the orbit normal, P_hat is the periapsis direction, and Q_hat is $\hat{Q} = \hat{c} \times \hat{P}$. Clearly these form an orthonormal basis.

To develop the appropriate formulae we apply the rotations of Eq. (10) to an identity matrix:

$$\begin{bmatrix} \hat{P}_x & \hat{Q}_x & \hat{c}_x \\ \hat{P}_y & \hat{Q}_y & \hat{c}_y \\ \hat{P}_z & \hat{Q}_z & \hat{c}_z \end{bmatrix} = \begin{bmatrix} \cos\Omega & -\sin\Omega & 0 \\ \sin\Omega & \cos\Omega & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos i & -\sin i \\ 0 & \sin i & \cos i \end{bmatrix} * \begin{bmatrix} \cos\omega & -\sin\omega & 0 \\ \sin\omega & \cos\omega & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\Omega & -\sin\Omega & 0 \\ \sin\Omega & \cos\Omega & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} \cos\omega & -\sin\omega & 0 \\ \cos i\sin\omega & \cos i\cos\omega & -\sin i \\ \sin i\sin\omega & \sin i\cos\omega & \cos i \end{bmatrix} =$$

$$\begin{bmatrix} (\cos\Omega\cos\omega - \sin\Omega\cos i\sin\omega) & (-\cos\Omega\sin\omega - \sin\Omega\cos i\cos\omega) & \sin\Omega\sin i \\ (\sin\Omega\cos\omega + \cos\Omega\cos i\sin\omega) & (-\sin\Omega\sin\omega + \cos\Omega\cos i\cos\omega) & -\cos\Omega\sin i \\ \sin i\sin\omega & \sin i\cos\omega & \cos i \end{bmatrix} \tag{B-16}$$

```
P_hat=zeros(3,2);Q_hat=P_hat;c_hat=P_hat;
P_hat(1,:)=clan.*carg_peri-slan.*cincl.*sarg_peri;
P_hat(2,:)=slan.*carg_peri+clan.*cincl.*sarg_peri;
P_hat(3,:)=sincl.*sarg_peri;
Q_hat(1,:)=-clan.*sarg_peri-slan.*cincl.*carg_peri;
Q_hat(2,:)=-slan.*sarg_peri+clan.*cincl.*carg_peri;
Q_hat(3,:)=sincl.*carg_peri;
c_hat=cross(P_hat,Q_hat,1);
norm(P_hat(:,1)),norm(P_hat(:,2)),
norm(Q_hat(:,1)),norm(Q_hat(:,2)),
norm(c_hat(:,1)),norm(c_hat(:,2))

ans =
     1
ans =
     1
ans =
     1
```

```
ans =
     1
ans =
     1
ans =
     1
```

Now that these have been calculated, we can directly determine positions and velocities. We use the formulae given in [Bond and Allman 1996, 45]:

$$\vec{r}(t) = a(\cos E(t) - e)\hat{P} + \sqrt{ap}\sin E(t)\hat{Q} \\ = (a(\cos E(t) - e))\hat{P} + \left(a\sqrt{1-e^2}\sin E(t)\right)\hat{Q} \tag{B-17}$$

and

$$\vec{v}(t) = \frac{\sqrt{\mu a}}{r}\sin E(t)\hat{P} + \frac{\sqrt{\mu p}}{r}\cos E(t)\hat{Q} \tag{B-18}$$

This last result can be restated in terms of the orbital period by making use of the relationship,

$$\frac{2\pi}{T_{period}} = \sqrt{\frac{\mu}{a^3}} \tag{B-19}$$

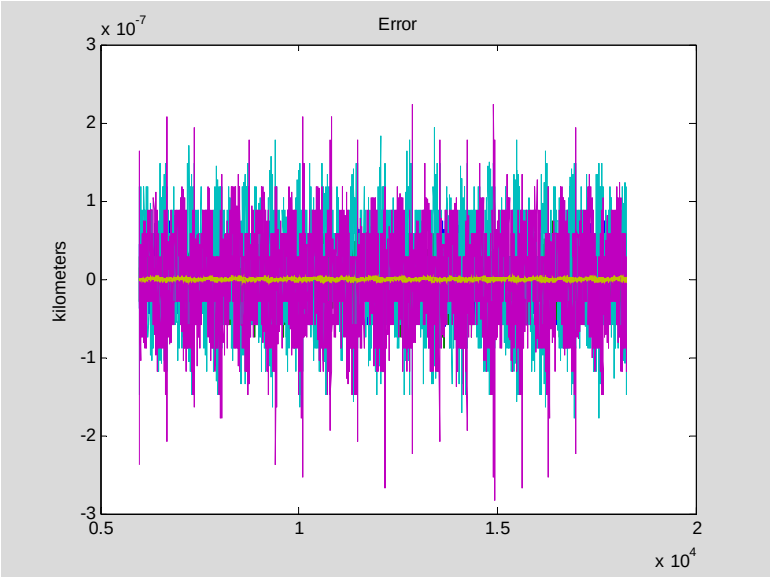
Then

$$\vec{v}(t) = \left(\frac{2\pi a^2 \sin E(t)}{r(t)T_{period}}\right)\hat{P} + \left(\frac{2\pi a^2\sqrt{1-e^2}\cos E(t)}{r(t)T_{period}}\right)\hat{Q} \tag{B-20}$$

```
XYZeXYZm_2=zeros(size(XYZeXYZm));
VXYZeXYZm=zeros(size(XYZeXYZm));
XYZeXYZm_2(:,1:3)=kmperAU*a_AU(1)*((cos(E(:,1))-ecc(:,1))
*P_hat(:,1)'+sqrt(1-ecc(:,1).^2) .*sin(E(:,1))*Q_hat(:,1)');
XYZeXYZm_2(:,4:6)=kmperAU*a_AU(2)*((cos(E(:,2))-ecc(:,2))*P_hat(:,2)'
+sqrt(1-ecc(:,2).^2) .*sin(E(:,2))*Q_hat(:,2)');
VXYZeXYZm(:,1:3)=2*pi*kmperAU^2*a_AU(1)^2./Tp_yrs(1)/daysperyear/24/360
0* (-sin(E(:,1))./r(:,1)*P_hat(:,1)'+sqrt(1-eccentricity(1)^2)
*cos(E(:,1))./r(:,1)*Q_hat(:,1)');
VXYZeXYZm(:,4:6)=2*pi*kmperAU^2*a_AU(2)^2./Tp_yrs(2)/daysperyear/24/360
0* (-sin(E(:,2))./r(:,2)*P_hat(:,2)'+sqrt(1-eccentricity(2)^2)
*cos(E(:,2))./r(:,2)*Q_hat(:,2)');

max(max(abs(XYZeXYZm_2-XYZeXYZm))'),
```

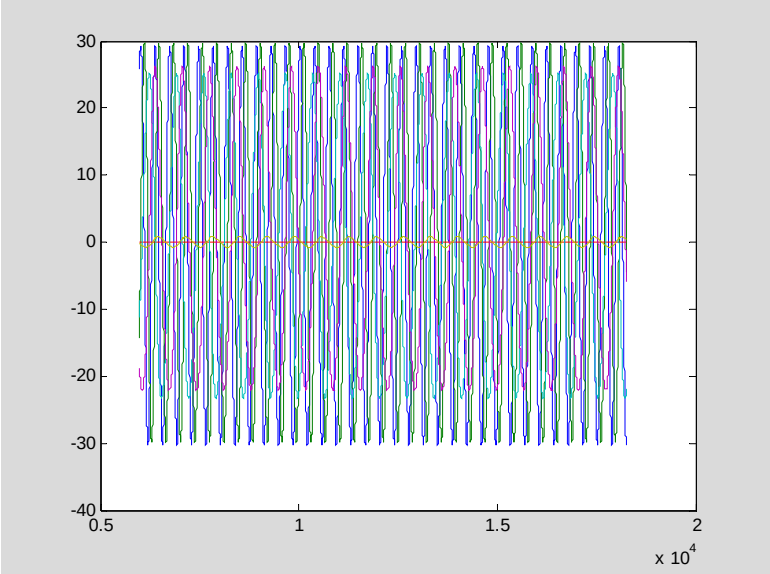
```
plot(t,XYZeXYZm_2-XYZeXYZm), title('Error'),ylabel('kilometers')
```

```
ans =
     2.83122062683105e-007
```

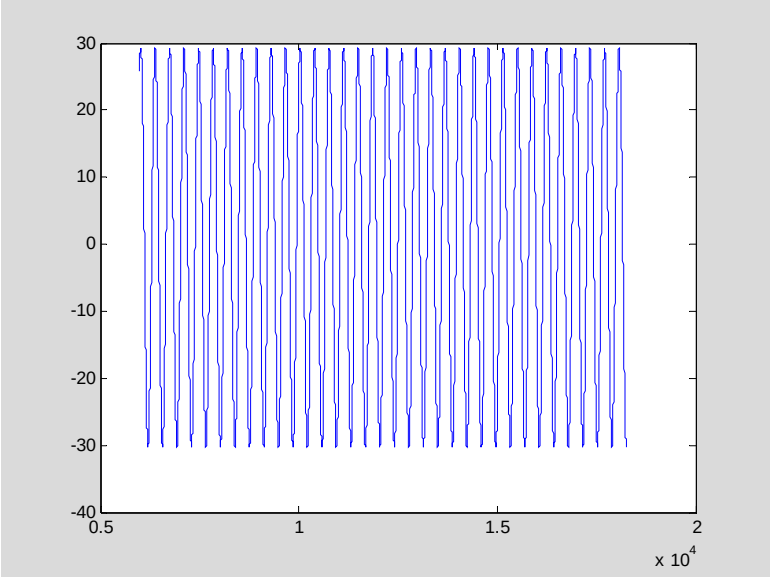


Next, show the velocities.

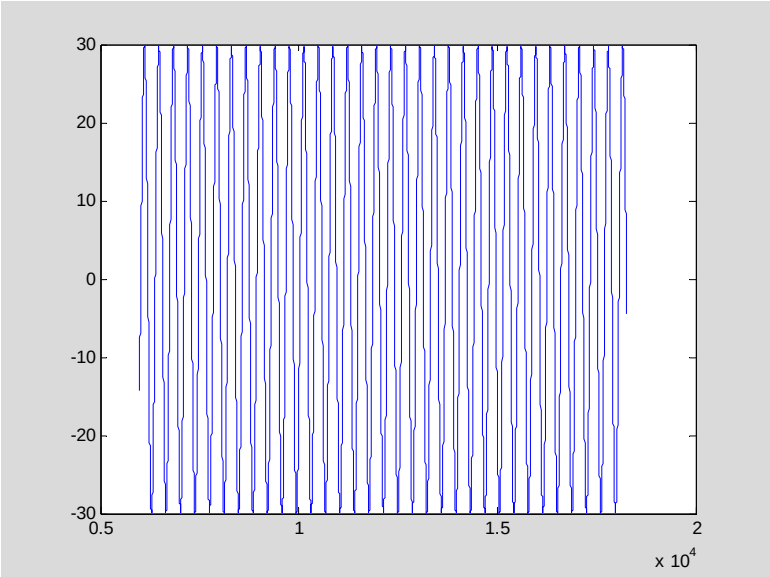
```
plot(t,VXYZeXYZm)
```



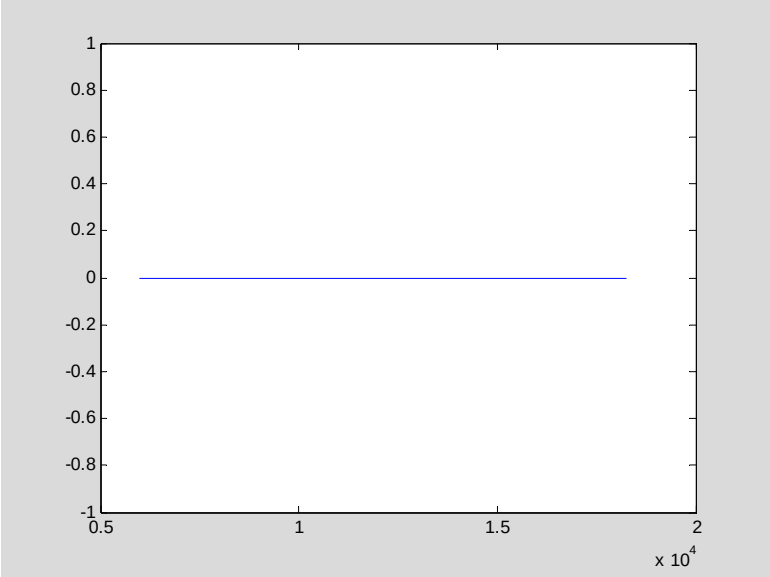

```
plot(t,VXYZeXYZm(:,1))
```



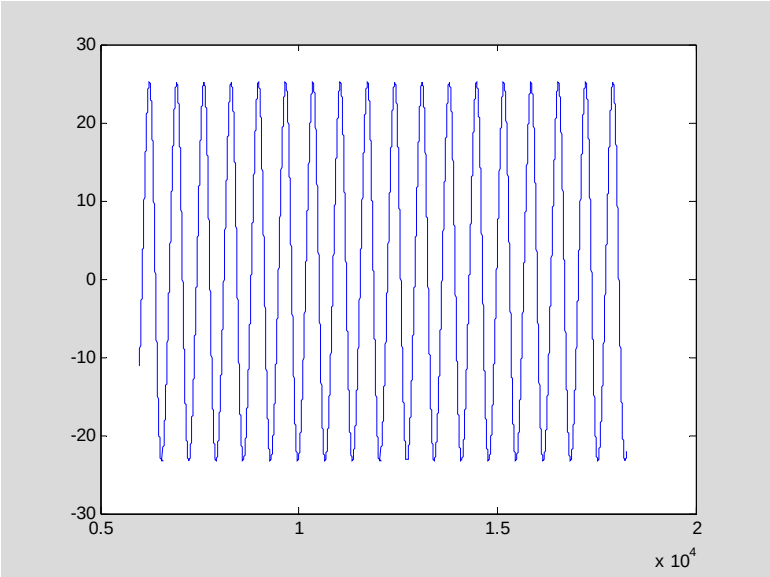

```
plot(t,VXYZeXYZm(:,2))
```



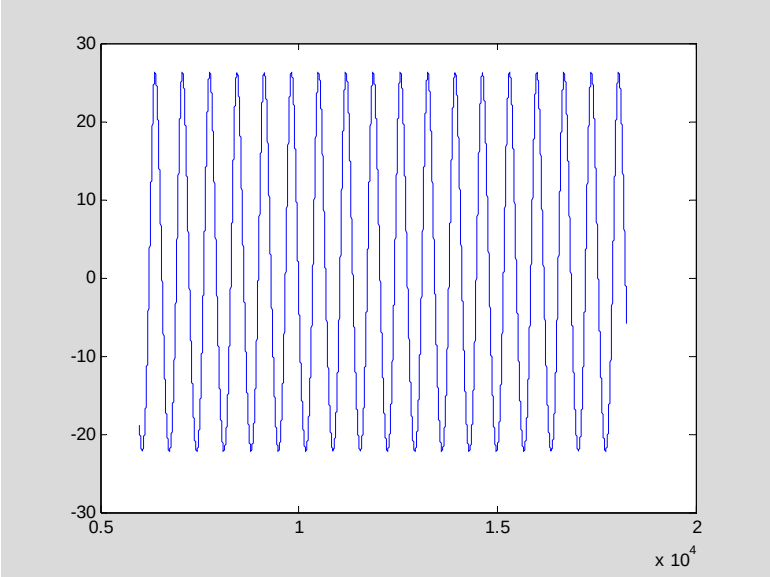

```
plot(t,VXYZeXYZm(:,3))
```



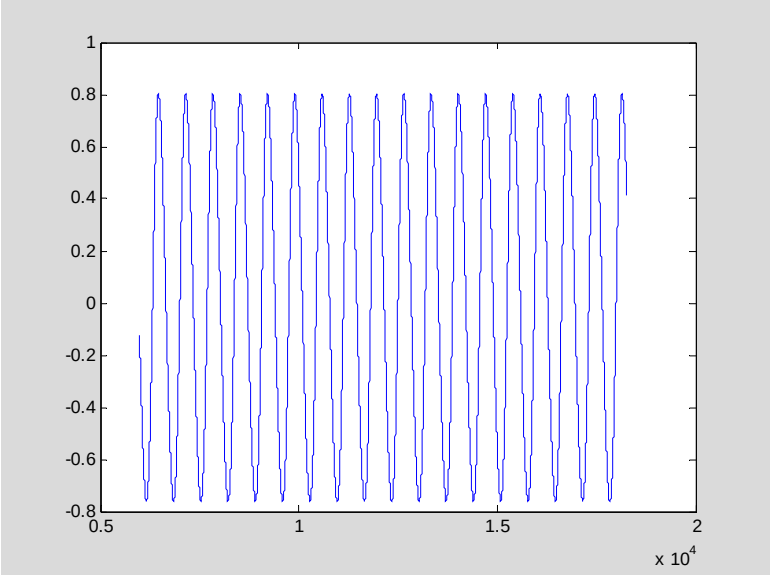

```
plot(t,VXYZeXYZm(:,4))
```



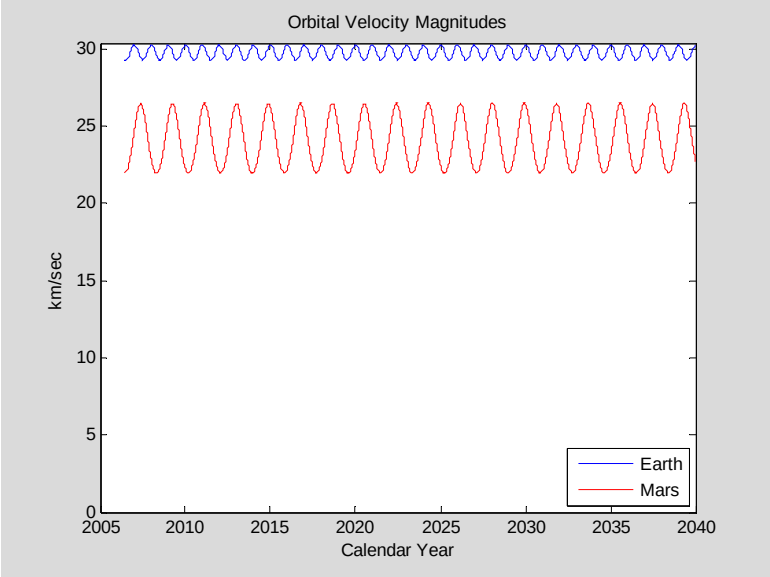

```
plot(t,VXYZeXYZm(:,5))
```




```
plot(t,VXYZeXYZm(:,6))
```




```
plot(t+t_epoch,sqrt(VXYZeXYZm(:,1).^2+VXYZeXYZm(:,2).^2+VXYZeXYZm(:,3).
^2),'b',t+t_epoch,sqrt(VXYZeXYZm(:,4).^2+VXYZeXYZm(:,5).^2+VXYZeXYZm(:,
6).^2),'r'),  datetick('x'), ylim([0 30.4]), ylabel('km/sec'),
xlabel('Calendar Year'), title('Orbital Velocity Magnitudes'),
legend('Earth','Mars','Location','SouthEast')
```



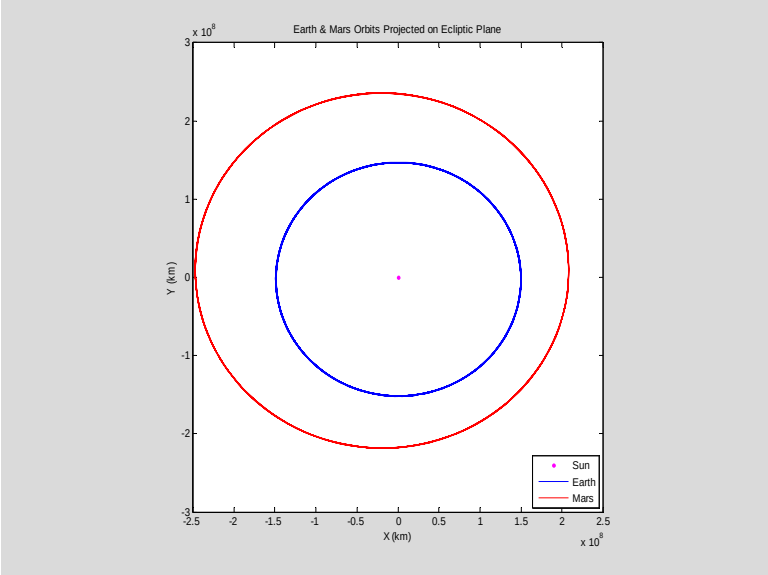
At this point, the script algorithms for calculating positions and velocities of Earth and Mars as functions of time have been debugged. Next these algorithms are commited to an M-file (i.e., a MATLAB subroutine).

```
function  [rearth,rmars,vearth,vmars]=EarthMars(t)
% Calculates Earth & Mars Position & Velocity Vectors for set of times.
% Coordinate System is standard Heliocentric Cartesian, units km & km/sec
% t is (1 by ) row vector of times using the standard MATLAB time system.
% Outputs are (3 by n) matrices whose columns are vectors at corresponding times.
% Algorithm uses orbital data valid for epoch date at start of calendar year 1990.
%  Coded by Robert Woolley, May 2006
%
daysperyear=365.242191;
kmperAU=149597870.;
Tp_yrs=[1.00004  1.880932];
incl_deg=[0.  1.849736];
a_AU=[1.00000  1.523688];
eccentricity=[0.016713  0.093396];
longt_asc_node_deg=[0  49.480308];
longt_peri_deg=[102.768413  335.874939];
longt_epoch_deg=[99.403308  240.739474];
t_epoch=datenum('00-Jan-1990');
lan=longt_asc_node_deg*pi/180.;
%
trueanomaly_epoch=[longt_epoch_deg-longt_peri_deg]*pi/180.;
E_epoch=acos((eccentricity+cos(trueanomaly_epoch))./(1+eccentricity.*cos(trueanomaly
_epoch)));
M0=eccentricity.*sin(E_epoch)-E_epoch;
```

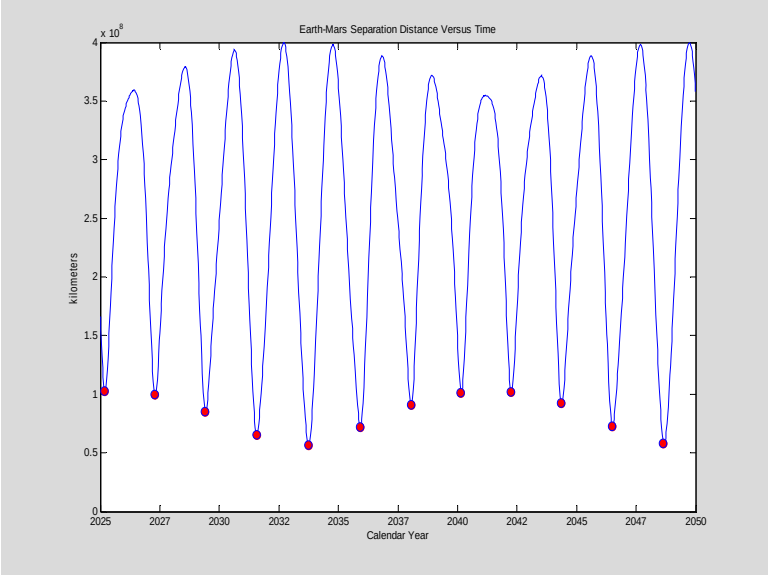
```
p=kmperAU*a_AU.*(1-eccentricity.^2);
arg_peri=(longt_peri_deg-longt_asc_node_deg)*pi/180.;
carg_peri=cos(arg_peri);sarg_peri=sin(arg_peri);
lan=longt_asc_node_deg*pi/180.;
clan=cos(lan);slan=sin(lan);
incl=incl_deg*pi/180.;
cincl=cos(incl);sincl=sin(incl);
% Iterative solution of Kepler's equation:
ecc=ones(size(t'))*eccentricity;
M=ones(size(t'))*M0+2*pi*t'/daysperyear*(1./Tp_yrs);
E=M;
for i=1:10;
 delta=E-ecc.*sin(E)-M;
  if abs(max(max(delta)'))<10*eps
   break
  end
 E=E-delta./(1-ecc.*cos(E));
end
phi=atan2(sqrt(1-ecc.^2).*sin(E)./(1-ecc.*cos(E)),(cos(E)-ecc)./(1-ecc.*cos(E)));
r=ones(size(t'))*p./(1+ecc.*cos(phi));
%
P_hat=zeros(3,2);Q_hat=P_hat;
P_hat(1,:)=clan.*carg_peri-slan.*cincl.*sarg_peri;
P_hat(2,:)=slan.*carg_peri+clan.*cincl.*sarg_peri;
P_hat(3,:)=sincl.*sarg_peri;
Q_hat(1,:)=-clan.*sarg_peri-slan.*cincl.*carg_peri;
Q_hat(2,:)=-slan.*sarg_peri+clan.*cincl.*carg_peri;
Q_hat(3,:)=sincl.*carg_peri;
%
rearthmars=zeros(6,size(t,2));
vearthmars=rearthmars;
rearth=zeros(3,size(t,2));
rmars=rearth;
vearth=rearth;
vmars=rearth;
%
rearth=(kmperAU*a_AU(1)*((cos(E(:,1))-ecc(:,1))*P_hat(:,1)' +sqrt(1-ecc(:,1).^2)
.*sin(E(:,1))*Q_hat(:,1')))';
rmars=(kmperAU*a_AU(2)*((cos(E(:,2))-ecc(:,2))*P_hat(:,2)' +sqrt(1-ecc(:,2).^2)
.*sin(E(:,2))*Q_hat(:,2')))';
vearth=(2*pi*kmperAU^2*a_AU(1)^2./Tp_yrs(1)/daysperyear/24/3600* (-
sin(E(:,1))./r(:,1)*P_hat(:,1)'+sqrt(1-eccentricity(1)^2) *cos(E(:,1))./r(:,1)*Q_hat(:,1')))';
vmars=(2*pi*kmperAU^2*a_AU(2)^2./Tp_yrs(2)/daysperyear/24/3600* (-
sin(E(:,2))./r(:,2)*P_hat(:,2)'+sqrt(1-eccentricity(2)^2) *cos(E(:,2))./r(:,2)*Q_hat(:,2')))';
```

The test for this code follows:

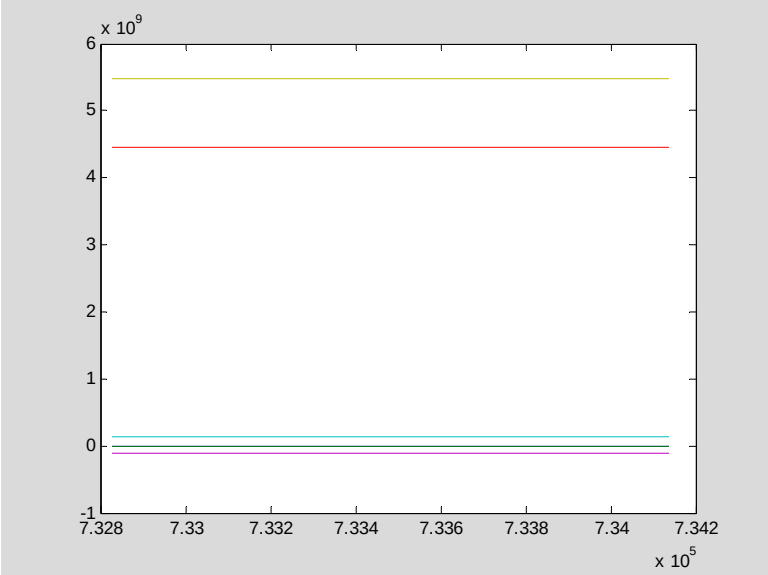
```
clear;

clf;t=linspace(datenum('1-Jan-2025'),datenum('31-dec-2049'),1000);
[rearth,rmars,vearth,vmars]=EarthMars(t);plot(0,0,'m.',rearth(1,:),rear
th(2,:),'b',rmars(1,:),rmars(2,:),'r'),ylim([-3e8 3e8]),xlabel('X
(km)'),ylabel('Y (km)'),title('Earth & Mars Orbits Projected on
Ecliptic Plane'), legend('Sun','Earth','Mars','Location','SouthEast'),
axis square
```




```
D_e_m=zeros(size(t));
for i=1:size(t,2);
D_e_m(1,i)=norm(rearth(:,i)-rmars(:,i));end,
III=find(diff(diff(D_e_m)<0)<0);Topposition=t(III);
plot(t',D_e_m',Topposition,D_e_m(III),'o','MarkerEdgeColor','b','Marker
FaceColor','r','MarkerSize',8),datetick('x'),xlabel('Calendar
Year'),ylabel('kilometers'),title('Earth-Mars Separation Distance
Versus Time'),ylim([0 4e8])
```




```
angmom_earth=cross(rearth,vearth);angmom_mars=cross(rmars,vmars);plot(t
,angmom_earth,t,angmom_mars)
```




```
ylim([0 4.5e8]),
```

The Lambert problem, also sometimes called the Gauss problem, is to find a free trajectory in a central gravitational field transferring between two specified locations in a specified time interval. That problem with the Sun as the central body is a main component of the problem of transferring between Earth and Mars, with the remaining part of the problem being the two terminal portions of the trajectory during which the planetary gravitational fields dominate.

[Bond and Allman 1996, 58-88]] and separately [Bate, Mueller and White 1971, 177-275] present derivations of a solution to the Lambert-Gauss problem using "universal variables". This solution makes use of versions of "Stumpff" functions, a class of functions defined by the following infinite series:

$$c_n(z) \equiv \sum_{k=0}^{\infty} (-1)^k \frac{z^k}{(2k+n)!} \tag{B-21}$$

where n is any nonnegative integer, $n \in \{0,1,2,3,...\}$.

Of this class of functions, only two appeared in the universal variables formulation. They are renamed in both references as follows:

$$S(z) \equiv c_3(z) \equiv \frac{1}{3!} - \frac{z}{5!} + \frac{z^2}{7!} - \frac{z^3}{9!} + ... \tag{B-22}$$

$$C(z) \equiv c_2(z) \equiv \frac{1}{2!} - \frac{z}{4!} + \frac{z^2}{6!} - \frac{z^3}{8!} + ... \tag{B-23}$$

As stated by both books, these can also be evaluated via closed form expressions as follows:

$$S(z) \equiv \begin{cases} \dfrac{\sqrt{z} - \sin\sqrt{z}}{\left(\sqrt{z}\right)^3} & \text{if } z > 0 \\ \dfrac{1}{3!} & \text{if } z = 0 \\ \dfrac{\sinh\sqrt{-z} - \sqrt{-z}}{\left(\sqrt{-z}\right)^3} & \text{if } z < 0 \end{cases} \tag{B-24}$$

$$C(z) \equiv \begin{cases} \dfrac{1 - \cos\sqrt{z}}{z} & \text{if } z > 0 \\ \dfrac{1}{2!} & \text{if } z = 0 \\ \dfrac{\cosh\sqrt{-z} - 1}{-z} & \text{if } z < 0 \end{cases} \tag{B-25}$$

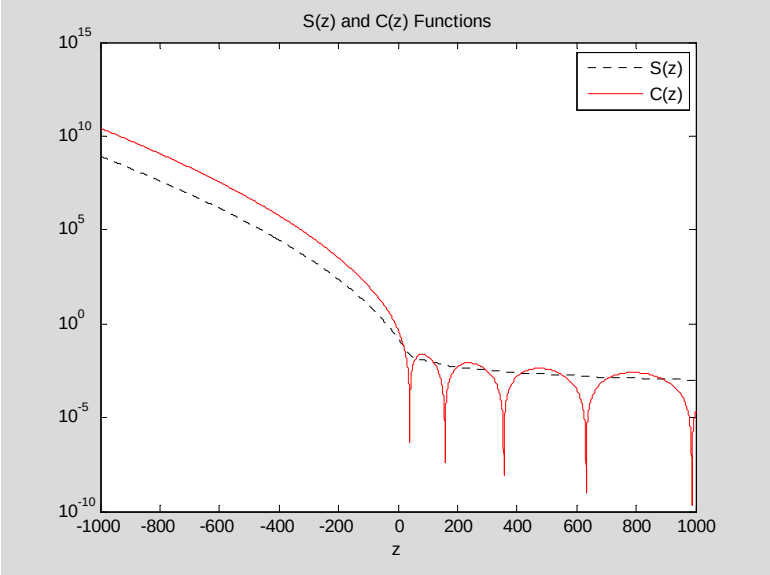
The derivatives of these functions are also given by [Bond and Allman 1996]] as

$$\frac{dC(z)}{dz} = \frac{1}{2z}\left[1 - zS(z) - 2C(z)\right] \quad \text{(B-26)}$$

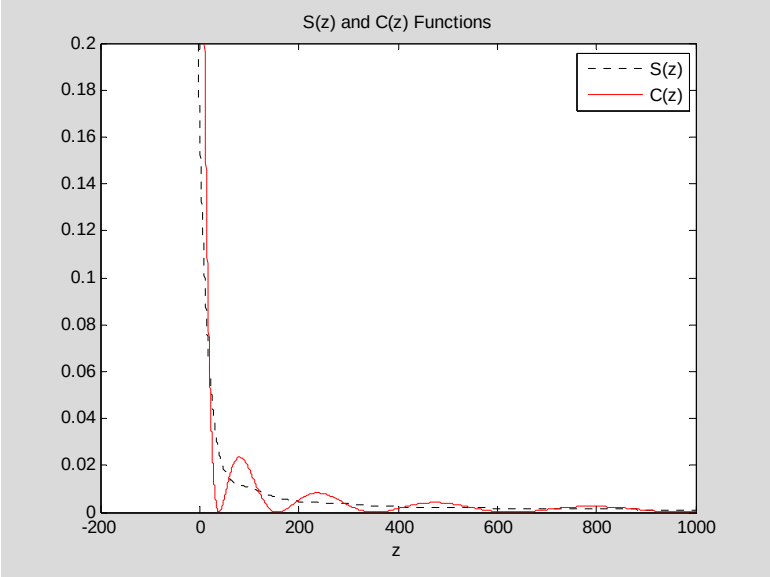
$$\frac{dS(z)}{dz} = \frac{1}{2z}\left[C(z) - 3S(z)\right] \quad \text{(B-27}$$

Approximate plots of these functions follow:

```
NF=10001;z=linspace(-1000,1000,NF);S=zeros(1,NF);C=S;
for i=1:NF;
 if z(i)<0
  zz=sqrt(-z(i));
  C(i)=(cosh(zz)-1)/zz^2;
  S(i)=(sinh(zz)-zz)/zz^3;
 elseif z(i)>0
  zz=sqrt(z(i));
  C(i)=(1-cos(zz))/zz^2;
  S(i)=(zz-sin(zz))/zz^3;
 else
  C(i)=1/2;
  S(i)=1/6;
 end
end
semilogy(z,S,':k',z,C,'r'),
legend('S(z)','C(z)'),xlabel('z'),title('S(z) and C(z) Functions')
```




```
plot(z,S,':k',z,C,'r'),ylim([0 0.2]),
legend('S(z)','C(z)'),xlabel('z'),title('S(z) and C(z) Functions')
```



Note that C(z) actually becomes zero at $z = (2\pi m)^2$ for each positive integer, n, a fact slightly missed by the grid of data values used in the above plots.

The solution to the Lambert-Gauss problem requires finding a value of "z" which satisfies a transcendental equation, $f_{(\vec{r},\vec{r}_0,\Delta t)}(z) = 0$. Both books use identical nomenclature in defining this function in terms of a parameter which they call A, and two auxiliary variables, x and y. Here, I have redefined the parameter and variables very slightly, replacing A by B and replacing x and y by xp and yp, with the purpose of using dimensionless parameters and variables wherever possible.

Stated explicitly, the function here is as follows:

$$f_{(\vec{r},\vec{r}_0,\Delta t)}(z) = \sqrt{\frac{\mu}{(r_0+r)^3}}\left[\tilde{\Delta}t(z) - \Delta t\right] \tag{B-28}$$

where we employ the "time-of-flight" function,

$$\tilde{\Delta}t(z) = \sqrt{\frac{(r_0+r)^3}{\mu}}\sqrt{1 - B\left(\frac{1-zS(z)}{\sqrt{C(z)}}\right)}\left(B + \frac{S(z)}{(C(z))^{3/2}}\left(1 - B\left(\frac{1-zS(z)}{\sqrt{C(z)}}\right)\right)\right) \tag{B-29}$$

and where the single dimensionless positive parameter, B, is determined from the specified problem conditions:

$$B = \frac{1}{r_0+r}\sqrt{\frac{r_0 r}{1-\cos\Delta\phi}}\sin\Delta\phi \tag{B-30}$$

It is important to understand that using this function of z should be restricted to a finite range of z values. Evaluation of the function's formula yields pure imaginary or complex values for z values less than a minimum value of z, with that minimum value of z depending on the actual value of the parameter, B. At that minimum value of z the value

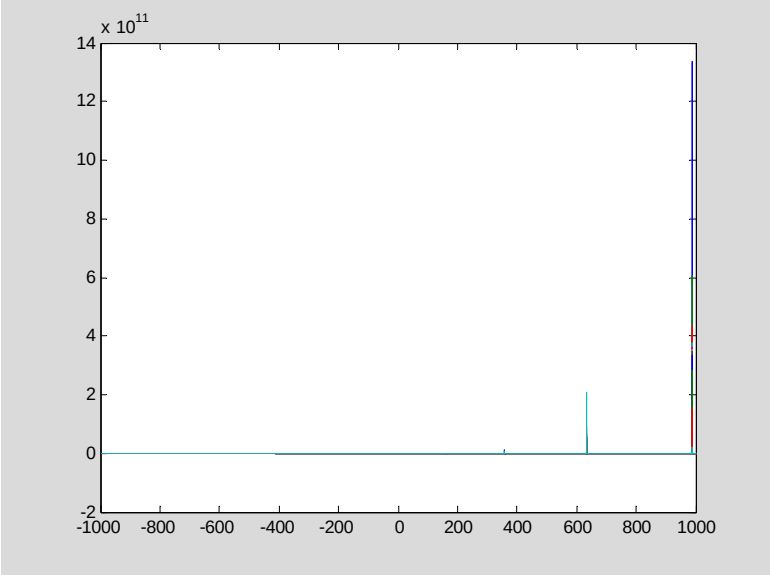
of the evaluated time-of-flight function is zero. As the z value is increased from that minimum value the time-of-flight function monotonically increases, approaching infinity as the z value approaches $+(2\pi)^2$. More positive values of z typically are not of interest since they correspond to transfer trajectories requiring multiple orbits to complete, in which the transfer trajectory returns to its starting location and perhaps its terminal location one or more times before meeting the end conditions. Thus, the typical range to consider for z values is $zmin(B)<z< +(2\pi)^2$

Relevant plots for different values of the parameter, B, follow:

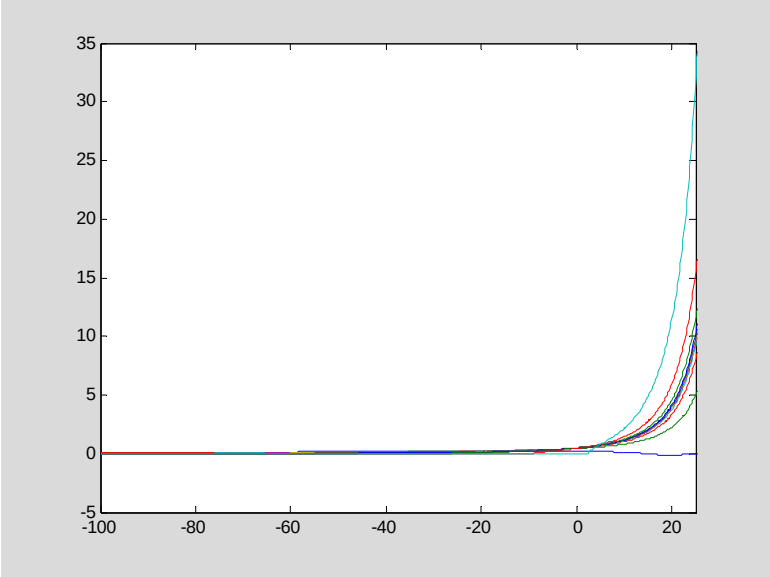
```
(1-z'.*S')./sqrt(C')
```

```
B=[-1 -.3 -.1 -.03 -.01 0 .01 .03 .1 .3 1];
yp=1-((1-z'.*S')./sqrt(C'))*B;  xp=sqrt(yp./(C'*ones(size(B))));
w=xp.^3.*(S'*ones(size(B)))+(ones(size(z'))*B).*sqrt(yp);
plot(z',w)
```

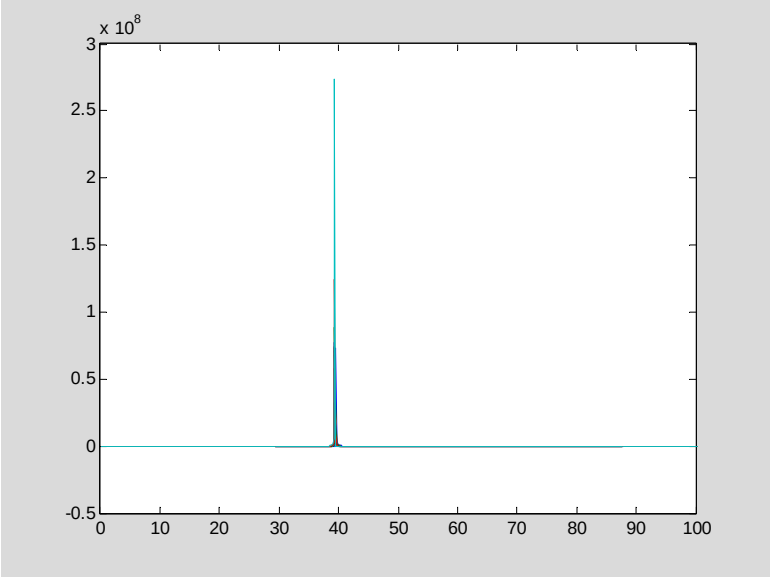
```
Warning: Imaginary parts of complex X and/or Y arguments ignored.
```



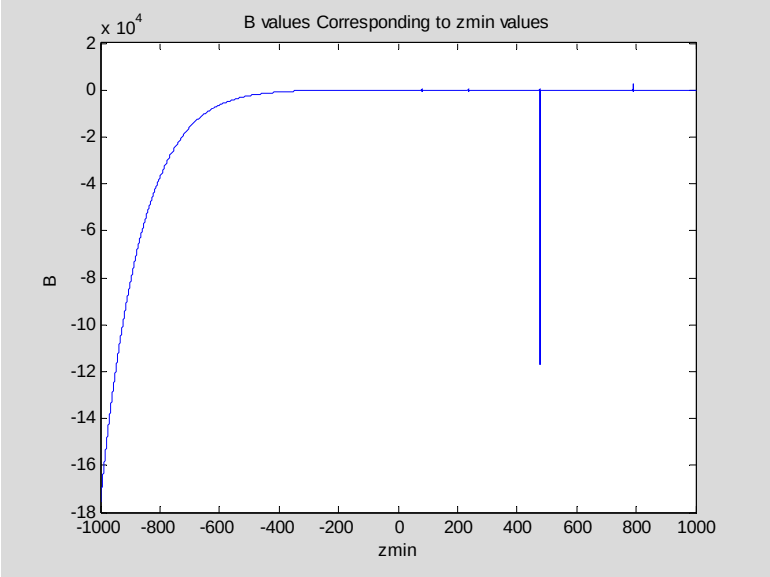

```
xlim([-100 25.])
```



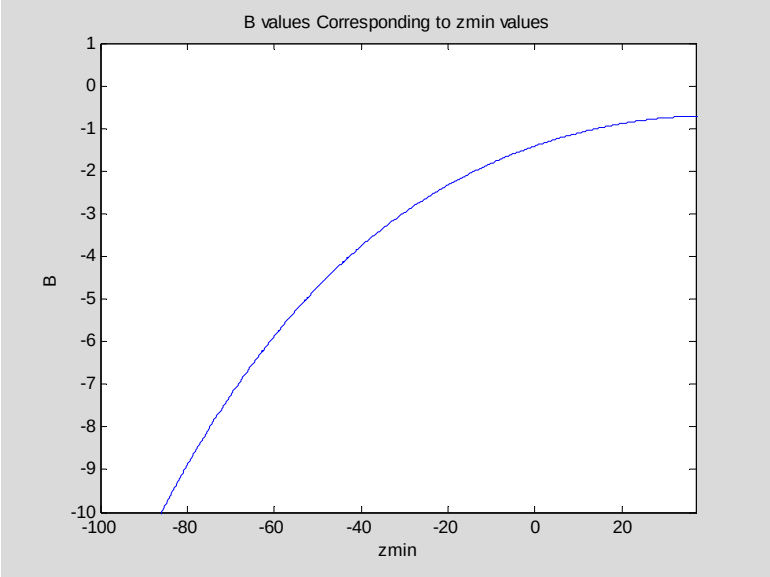

```
xlim([0 100])
```



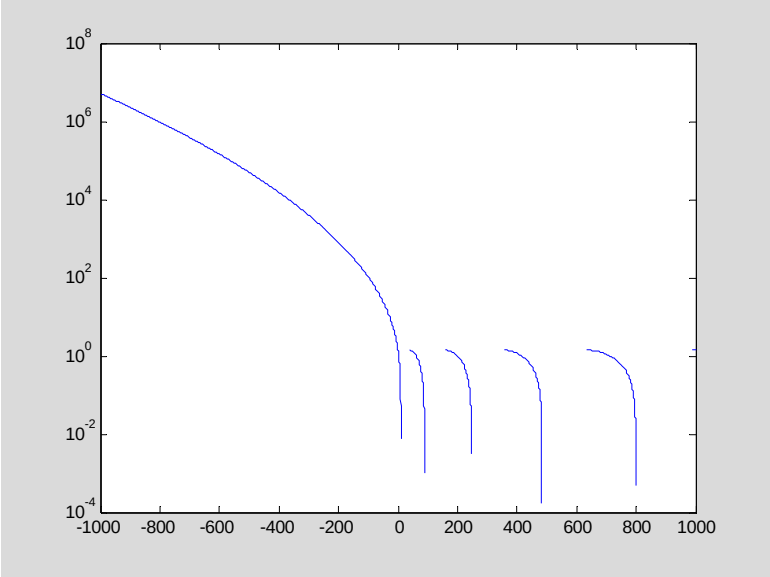

```
Bzmin=S.*sqrt(C)./(S-z.*S.^2-C.^2);plot(z,Bzmin),title('B values
Corresponding to zmin values'),xlabel('zmin'),ylabel('B')
```



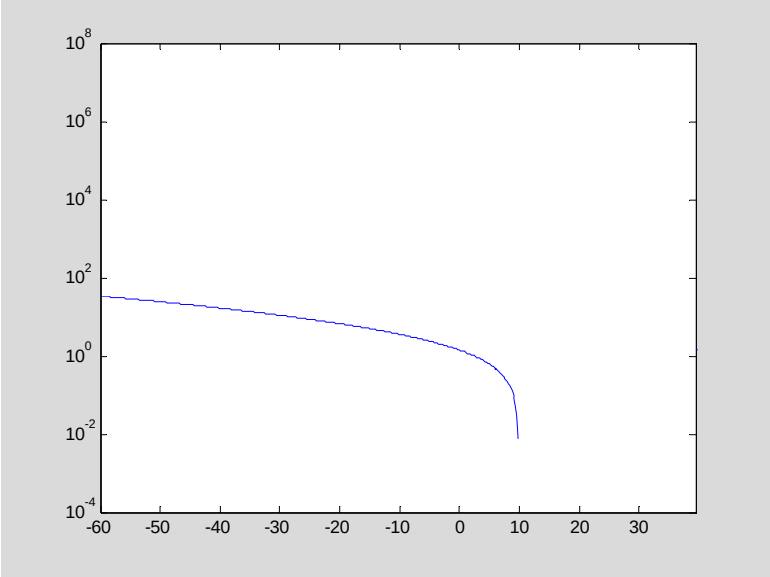

```
xlim([-100 37]),ylim([-10 1])
```



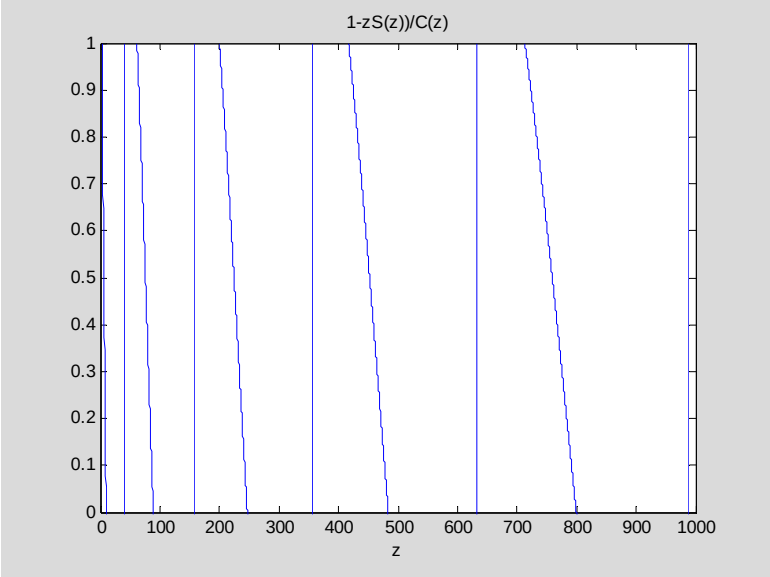

```
semilogy(z,(1-z'.*S')./sqrt(C'))
```



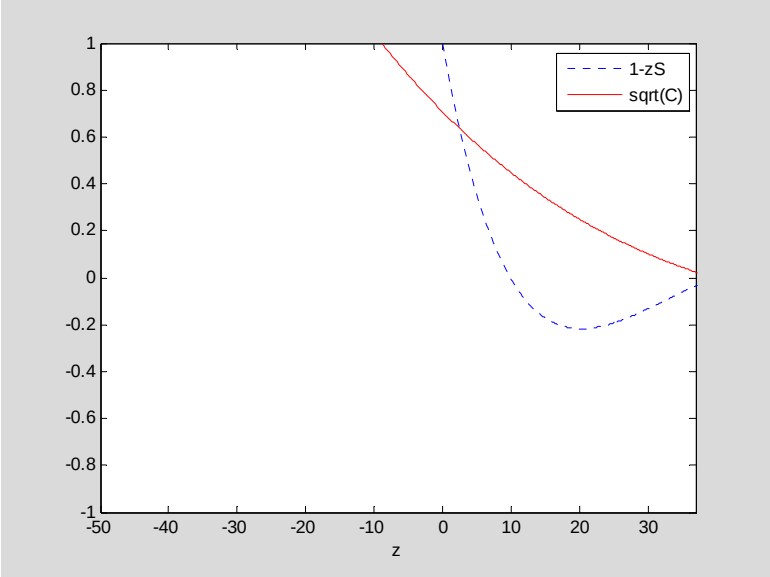

```
xlim([-60 (2*pi)^2])
```




```
plot(z,(1-z'.*S')./sqrt(C')),xlabel('z'),title('1-zS(z))/C(z)'),
ylim([0 1])
```




```
plot(z,(1-z'.*S'),'b:',z,sqrt(C),'r'),xlabel('z'),ylim([-1 1]),
```
xlim([-50 37]),legend('1-zS','sqrt(C)')



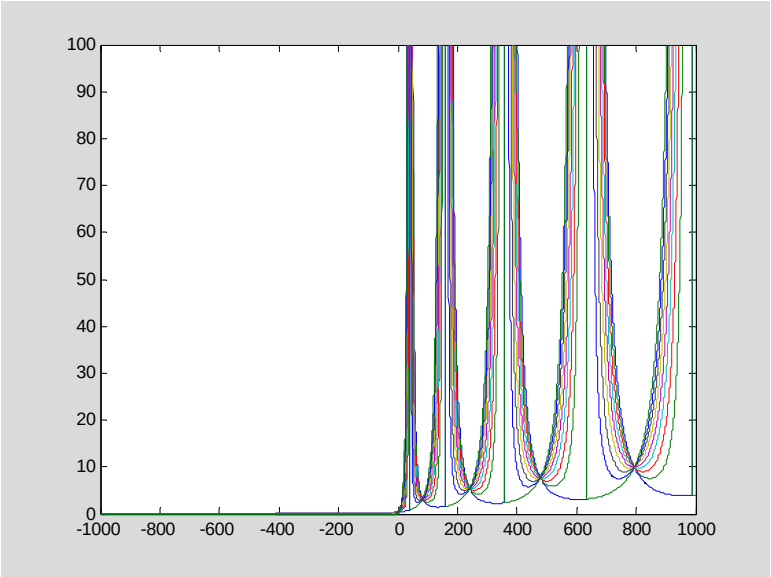
])

To calculate the normalized time-of-flight function conveniently, an m-file was coded as follows.

```
function [TOF,S,C,xp,yp]=Lambert(z,B);
% Normalized Time-of-flight calculated for modified
universal variables
% This is used in solution of the Lambert-Gauss problem.
% I/O defs:
```

```
%  z must be a (n by 1) column vector of z values, for
integer n>0.
%  B must be a (1 by m) row vector of B values, for integer
m>0.
%  TOF is then a (n by m) matrix of resulting normalized
time-of-flight values.
% Coded by Robert Woolley  May 2006
%
d=1e-10;
I=find(z<-d);J=find(-d<=z & z<=d);K=find(z>d);
zz=zeros(size(z));S=zz;C=zz;
zz(I)=sqrt(-z(I));zz(K)=sqrt(z(K));
C(I)=(cosh(zz(I))-1)./zz(I).^2;
S(I)=(sinh(zz(I))-zz(I))./zz(I).^3;
C(J)=1/2-z(J)/24+z(J).^2/720;
S(J)=1/6-z(J)/120+z(J).^2/5040;
C(K)=(1-cos(zz(K)))./zz(K).^2;
S(K)=(zz(K)-sin(zz(K)))./zz(K).^3;
yp=1-((1-z.*S)./sqrt(C))*B;
xp=sqrt(yp./(C*ones(size(B))));
TOF=zeros(size(z,1),size(B,2));
TOF=xp.^3.*(S*ones(1,size(B,2)))+(ones(size(z,1),1)*B).*sqr
t(yp);
```

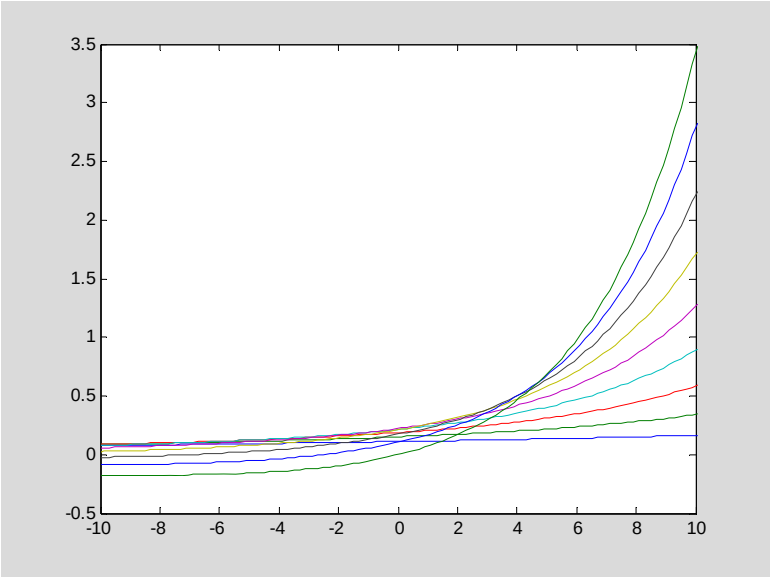
```
Warning: Imaginary parts of complex X and/or Y arguments ignored.
```




```
xlim([-50 37])
```

This m-file seems to be working well.

```
plot(z,Lambert(z,B).^2),xlim([-50 37]),ylim([-10 100])

xlim([-10 10]),ylim('auto')
```



The computational algorithm for the full Lambert problem's solution, based on the slightly different exposition by [Bond et al] follows.

Problem Definition:

Given $\vec{r}_0, \vec{r}, \Delta t = t - t_0,$

Find $\frac{d\vec{r}_0}{dt}, \frac{d\vec{r}}{dt}$

Algorithm for an intercept trajectory transfer orbit follows. Note that in general there are two alternative intercept trajectories, direct and retrograde (in the sense of the diretion of revolution around the sun). In most anticipated cases the direct intercept trajectory requires less change in velocity, although the retrograde intercept trajectory may perhaps require less change for special situations where the required velocity change is very large.

(1) Compute $r_0, r, \Delta\phi$

$r_0 = \|\vec{r}_0\|$

$r = \|\vec{r}\|$

$$\Delta\phi = \begin{cases} \cos^{-1}\left(\dfrac{\vec{r}_0 \bullet \vec{r}}{r_0 r}\right) & \text{if } \hat{k} \bullet (\vec{r}_0 \times \vec{r}) > 0 \\ 2\pi - \cos^{-1}\left(\dfrac{\vec{r}_0 \bullet \vec{r}}{r_0 r}\right) & \text{if } \hat{k} \bullet (\vec{r}_0 \times \vec{r}) < 0 \end{cases}$$ for a *direct* intercept trajectory,

and

$$\Delta\phi = \begin{cases} 2\pi - \cos^{-1}\left(\dfrac{\vec{r}_0 \bullet \vec{r}}{r_0 r}\right) & \text{if } \hat{k} \bullet (\vec{r}_0 \times \vec{r}) > 0 \\ \cos^{-1}\left(\dfrac{\vec{r}_0 \bullet \vec{r}}{r_0 r}\right) & \text{if } \hat{k} \bullet (\vec{r}_0 \times \vec{r}) < 0 \end{cases}$$ for a *retrograde* intercept trajectory.

where $\hat{k}$ is the ecliptic plane's unit-normal (on the north side).

The unit vector, $\hat{k}$, is used above because it is the earth's orbit normal direction, and is aligned with earth's orbital angular momentum vector. Since all planets orbit in planes close to earth's, it may be practically used for any transfer between them. However, it should theoretically be replaced by the angular momentum direction of the initial planet's orbit.

(2) Calculate $A = \sqrt{\dfrac{r_0 r}{1-\cos\Delta\phi}}\sin\Delta\phi$ or $B = \dfrac{1}{r_0 + r}\sqrt{\dfrac{r_0 r}{1-\cos\Delta\phi}}\sin\Delta\phi$

(Note that *B* can only range over the interval, $-\dfrac{\sqrt{2}}{2} \le B \le \dfrac{\sqrt{2}}{2}$)

(3) Iterate the following to solve for z. Start with guessed value of z.

(a) Calculate C=C(z) and S=S(z).

(b) Calculate y and x from

$y = r_0 + r - A\left[\dfrac{1-zS}{\sqrt{C}}\right]$ and $x = \sqrt{\dfrac{y}{C}}$

or

Calculate y_p and x_p from

$y_p = 1 - B\left[\dfrac{1-zS}{\sqrt{C}}\right] = \dfrac{y}{(r_0 + r)}$ and $x_p = \sqrt{\dfrac{y_p}{C}} = \dfrac{x}{\sqrt{(r_0 + r)}}$

(c) Calculate $\tilde{\Delta}t = \frac{1}{\sqrt{\mu}}\left[x^3 S + A\sqrt{y}\right]$ or $\tilde{\Delta}t = \sqrt{\frac{(r_0 + r)^3}{\mu}}\left[x_p^3 S + B\sqrt{y_p}\right]$

(d) Calculate $\delta = \tilde{\Delta}t - \Delta t$

(e) If $|\delta| <$ error tolerance then proceed to step (4), exiting this iteration.

(f) Otherwise, improve the estimate of z and repeat the present iteration.

(4) Calculate the following:

$$f = 1 - \frac{y}{r_0} = 1 - \frac{r_0 + r}{r_0} y_p$$

$$g = A\sqrt{\frac{y}{\mu}} = B\sqrt{\frac{y_p (r_0 + r)^3}{\mu}}$$

$$\dot{g} = 1 - \frac{y}{r} = 1 - \frac{r_0 + r}{r} y_p$$

(5) Calculate the initial and terminal velocities:

$$\vec{v}_0 = \frac{1}{g}\left[\vec{r} - f\vec{r}_0\right]$$

$$\vec{v} = \frac{1}{g}\left[\dot{g}\vec{r} - \vec{r}_0\right]$$

For the *direct* heliocentric transfer from Earth's location to Mars' location, this is implemented in the following MATLAB script.

```
radiuse=6378;radiusm=3380;
mus=1.327151544E11;mue=3.986e5;mum=4.305e4;
r=zeros(3,1);r0=r;
t_array=[datenum('1-November-2007')  datenum('27-February-
2008')];[rearth,rmars,vearth,vmars]=EarthMars(t_array)
r0=rearth(:,1),
r=rmars(:,2),
t=t_array(2);t0=t_array(1);
%
rmag=sqrt(dot(r,r));r0mag=sqrt(dot(r0,r0));
if dot([0 0 1],cross(r0,r))>0;
 deltaphi=acos(dot(r0,r)/(r0mag*rmag));
else
 deltaphi=2*pi-acos(dot(r0,r)/(r0mag*rmag));
end
B=sin(deltaphi)*sqrt(r0mag*rmag/(1-cos(deltaphi)))/(r0mag+rmag);
TOF0=24*3600*(t-t0)*sqrt(mus/(r0mag+rmag)^3),
TOF02=TOF0^2,
z=zeros(1,1);
```

```
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=2*z-1;
 end
end
zmin=z(1),
z=(2*pi)^2-.1;
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=z/2;
 end
end
zmax=2*z(1),
[z_soln,fval,exitflag]=fzero(@(z) Lambert(z,B(1)).^2-TOF02,[zmin
zmax]),
z=z_soln,
[TOF,S,C,xp,yp]=Lambert(z,B(1)),
f=1-(r0mag+rmag)/r0mag*yp;
gdot=1-(r0mag+rmag)/rmag*yp;
g=B*sqrt(yp)*sqrt((r0mag+rmag)^3/mus);
v0=(r-f*r0)/g
v=(gdot*r-r0)/g
vde_hyperbolicexcess=v0-vearth(:,1),
vde_he_norm=norm(vde_hyperbolicexcess),
vde_norm=sqrt(2*mue/(radiuse)+vde_he_norm^2),
vam_hyperbolicexcess=v-vmars(:,2),
vam_he_norm=norm(vam_hyperbolicexcess),
vam_norm=sqrt(2*mum/(radiusm+100)+vam_he_norm^2),


rearth =
          147292521.201715          -93293690.514889
          26555828.5544097          114057055.931532
                         0                         0
rmars =
          73523334.0725974         -150464848.703335
          215722719.460728          194744822.143563
          2721384.48999595          7780183.04869839
vearth =
         -5.77105904654614          -23.543403652262
          29.2061674104939         -18.9703501940611
                         0                         0
vmars =
         -22.0060066348209         -18.2485412500953
          9.87188464347986         -12.7551422456007
          0.747389877519146         0.180371633250415
r0 =
          147292521.201715
```

```
          26555828.5544097
                         0
r =
         -150464848.703335
          194744822.143563
          7780183.04869839
TOF0 =
         0.471513126070999
TOF02 =
         0.222324628057246
zmin =
     0
zmax =
          1.23057555013617
z_soln =
         0.898628910517223
fval =
     0
exitflag =
     1
z =
         0.898628910517223
TOF =
         0.471513126070999
S =
         0.159336333878151
C =
         0.463660884252302
xp =
          1.09118062377873
yp =
          0.55206949462649
v0 =
         -14.4546076303842
          36.1894389249014
          1.36040292082504
v =
         -28.7107827523228
        -0.817543117441966
         0.152846126016654
vde_hyperbolicexcess =
         -8.68354858383802
          6.98327151440747
          1.36040292082504
vde_he_norm =
          11.2258983230207
vde_norm =
           15.843388327965
vam_hyperbolicexcess =
         -10.4622415022275
          11.9375991281587
       -0.0275255072337608
vam_he_norm =
          15.8734220585574
vam_norm =
```

```
16.6345095256651
```

Without considering the effects of planetary gravitational fields, the above implements the *direct* Lambert solution for heliocentric transfer from earth's location to mars' locations at specified times. We need to implement the direct transfer from mars to earth's location at specified boundary times. We also need to implement the *retrograde* Lambert solutions in both directions.

A slight modification to the script, i.e., changing the direction of the inequality on the triple product appearing in the tenth line, yields the *retrograde* heliocentric transfer from Earth's location to Mars' location.

```
radiuse=6378;radiusm=3380;
mus=1.327151544E11;mue=3.986e5;mum=4.305e4;
r=zeros(3,1);r0=r;
t_array=[datenum('1-November-2007')  datenum('1-July-2008')];
[rearth,rmars,vearth,vmars]=EarthMars(t_array)
r0=rearth(:,1)
r=rmars(:,2)
t=t_array(2);t0=t_array(1);
%
rmag=norm(r),r0mag=norm(r0);
if dot([0 0 1],cross(r0,r))<0;
 deltaphi=acos(dot(r0,r)/(r0mag*rmag));
else
 deltaphi=2*pi-acos(dot(r0,r)/(r0mag*rmag));
end
B=sin(deltaphi)*sqrt(r0mag*rmag/(1-cos(deltaphi)))/(r0mag+rmag);
TOF0=24*3600*(t-t0)*sqrt(mus/(r0mag+rmag)^3);
TOF02=TOF0^2;
NTOFerr=@(z,B) Lambert(z(1),B(1))^2-TOF02;
z=zeros(1,1);
z=zeros(1,1);
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=2*z-1;
 end
end
zmin=z(1),
z=(2*pi)^2-.1;
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
```

```
  z=z/2;
 end
end
zmax=2*z(1),
[z_soln,fval,exitflag]=fzero(@(z) Lambert(z,B(1)).^2-TOF02,[zmin
zmax]),
z=z_soln;
[TOF,S,C,xp,yp]=Lambert(z,B(1))
f=1-(r0mag+rmag)/r0mag*yp;
gdot=1-(r0mag+rmag)/rmag*yp;
g=B*sqrt(yp)*sqrt((r0mag+rmag)^3/mus);
v0=(r-f*r0)/g
v=(gdot*r-r0)/g,
vde_hyperbolicexcess=v0-vearth(:,1),
vde_he_norm=norm(vde_hyperbolicexcess),
vde_norm=sqrt(2*mue/(radiuse+300)+vde_he_norm^2),
vam_hyperbolicexcess=v-vmars(:,2),
vam_he_norm=norm(vam_hyperbolicexcess),
vam_norm=sqrt(2*mum/(radiusm+100)+vam_he_norm^2),

rearth =
          147292521.201715         -45354088.6904164
          26555828.5544097         -144812826.344405
                         0                         0
rmars =
          73523334.0725974         -246162443.428591
          215722719.460728          -11557574.074639
          2721384.48999595          5800844.23000556
vearth =
         -5.77105904654614          27.9417221210745
          29.2061674104939          -9.0132013599925
                         0                         0
vmars =
         -22.0060066348209          2.04909824339266
          9.87188464347986         -22.1371172128038
         0.747389877519146        -0.514797865778934
r0 =
          147292521.201715
          26555828.5544097
                         0
r =
         -246162443.428591
          -11557574.074639
          5800844.23000556
rmag =
          246501878.019227
zmin =
     0
zmax =
          9.84460440108936
z_soln =
           8.8959539084112
fval =
     4.44089209850063e-016
exitflag =
```

```
     1
TOF =
         0.969972134632687
S =
          0.10644396027001
C =
         0.223403656206305
xp =
          2.12113141377819
yp =
          1.00513738920485
v0 =
          1.58030762262947
          -32.779002739751
         -5.84327627649743
v =
         -4.23188306971843
           19.585270462473
          3.59607817098754
vde_hyperbolicexcess =
          7.35136666917561
         -61.9851701502449
         -5.84327627649743
vde_he_norm =
          62.6924858982555
vde_norm =
          63.6374484647417
vam_hyperbolicexcess =
          -6.2809813131111
          41.7223876752767
          4.11087603676648
vam_he_norm =
          42.3923066295217
vam_norm =
          42.6831236050238
```

Finally, we need to implement Fimple's midcourse plane change trajectories for situations where $\Delta\phi$>90 degrees, and do it for both transfer directions. To calculate this two step heliocentric trajectory, we first calculate a Lambert transfer to the *image* of the destination planet *rotated onto the orbital plane of the originating planet.* Note that this rotational projection preserves the planet's distance from the sun and aangular position in the orbit plane. We then identify the point in the trajectory which is 90 degrees from the image, and identify the time when that point is reached. Finally we calculate a Lambert trajectory from that point to the destination planet's actual final position.

The following MATLAB script calculates a Fimple direct transfer with plane change maneuver from Earth to Mars if the heliocentric angle between departure and arrival locations exceeds 90 degrees. For smaller angles it calculates a direct Lambert transfer.

```
radiuse=6378;radiusm=3380;
```

```
mus=1.327151544E11;mue=3.986e5;mum=4.305e4;
r=zeros(3,1);r0=r;
t_array=[datenum('1-November-2007')  datenum('1-July-2008')];
[rearth,rmars,vearth,vmars]=EarthMars(t_array),
r0=rearth(:,1),
r=rmars(:,2),
t=t_array(2);
t0=t_array(1);
%
rmag=norm(r),
r0mag=norm(r0);
if dot([0 0 1]',cross(r0,r))>0,
 deltaphi=acos(dot(r0,r)/(r0mag*rmag)),
else
 deltaphi=2*pi-acos(dot(r0,r)/(r0mag*rmag)),
end
midcourse_deltav=zeros(3,1);
if deltaphi<pi/2
%
B=sin(deltaphi)*sqrt(r0mag*rmag/(1-cos(deltaphi)))/(r0mag+rmag);
TOF0=24*3600*(t-t0)*sqrt(mus/(r0mag+rmag)^3);
TOF02=TOF0^2;
NTOFerr=@(z,B) Lambert(z(1),B(1))^2-TOF02;
z=zeros(1,1);
z=zeros(1,1);
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=2*z-1;
 end
end
zmin=z(1),
z=(2*pi)^2-.1;
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=z/2;
 end
end
zmax=2*z(1),
[z_soln,fval,exitflag]=fzero(@(z) Lambert(z,B(1)).^2-TOF02,[zmin
zmax]),
z=z_soln;
[TOF,S,C,xp,yp]=Lambert(z,B(1));
f=1-(r0mag+rmag)/r0mag*yp;
gdot=1-(r0mag+rmag)/rmag*yp;
g=B*sqrt(yp)*sqrt((r0mag+rmag)^3/mus);
v0=(r-f*r0)/g
v=(gdot*r-r0)/g
vde_hyperbolicexcess=v0-vearth(:,1),
vde_he_norm=norm(vde_hyperbolicexcess),
```

```
vde_norm=sqrt(2*mue/(radiuse+300)+vde_he_norm^2),
vam_hyperbolicexcess=v-vmars(:,2),
vam_he_norm=norm(vam_hyperbolicexcess),
vam_norm=sqrt(2*mum/(radiusm+100)+vam_he_norm^2)
%
else
%
%  Rotate destination about mcc direction into source orbit plane
%   in order to form preliminary target
he=cross(rearth(:,1),vearth(:,1));he_hat=he/norm(he);
mcc_hat=cross(rmars(:,2),he_hat);mcc_hat=mcc_hat/norm(mcc_hat),
rtarget=cross(he_hat,mcc_hat)*norm(rmars(:,2)),
r=rtarget,
% True anomaly difference to target has changed.  Need to recalcualte.
if dot([0 0 1]',cross(r0,r))>0,
 deltaphi1=acos(dot(r0,r)/(r0mag*rmag)),
else
 deltaphi1=2*pi-acos(dot(r0,r)/(r0mag*rmag)),
end
B=sin(deltaphi1)*sqrt(r0mag*rmag/(1-cos(deltaphi1)))/(r0mag+rmag);
TOF0=24*3600*(t-t0)*sqrt(mus/(r0mag+rmag)^3);
TOF02=TOF0^2;
NTOFerr=@(z,B) Lambert(z(1),B(1))^2-TOF02;
z=zeros(1,1);
z=zeros(1,1);
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=2*z-1;
 end
end
zmin=z(1),
z=(2*pi)^2-.1;
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=z/2;
 end
end
zmax=2*z(1),
[z_soln,fval,exitflag]=fzero(@(z) Lambert(z,B(1)).^2-TOF02,[zmin
zmax]),
z=z_soln(1);
[TOF,S,C,xp,yp]=Lambert(z,B(1))
f=1-(r0mag+rmag)/r0mag*yp(1);
gdot=1-(r0mag+rmag)/rmag*yp(1);
g=B(1)*sqrt(yp(1))*sqrt((r0mag+rmag)^3/mus);
v0=(r-f*r0)/g
v=(gdot*r-r0)/g
vde_hyperbolicexcess=v0-vearth(:,1),
vde_he_norm=norm(vde_hyperbolicexcess),
```

```
vde_norm=sqrt(2*mue/(radiuse+300)+vde_he_norm^2),
%
%  FIND Fimple's MIDCOURSE CORRECTION POINT 90 degrees before rtarget.
%  -Find parameters of first part of transfer trajectory (pre-
midcourse)
E=dot(v0,v0)/2-mus/r0mag,
h=cross(r0,v0),hnorm=sqrt(dot(h,h)),h_hat=h/hnorm,
P=-(cross(h,v0)+r0*mus/r0mag),
Pnorm=sqrt(dot(P,P)),
P_hat=P/Pnorm,
Q_hat=cross(h_hat,P_hat),
e=Pnorm/mus,
p=dot(h,h)/mus,
dot(r0,P_hat),
dot(r0,Q_hat),
phi0=atan2(dot(r0,Q_hat),dot(r0,P_hat)),
%  Find mid-course correction (mc) location and pre-burn velocity.
phimcc=phi0+deltaphi1-pi/2.,
rmcc_norm=p/(1+e*cos(phimcc)),
rmcc=rmcc_norm*(P_hat*cos(phimcc)+Q_hat*sin(phimcc)),
v1mcc=sqrt(mus/p)*(-sin(phimcc)*P_hat+(e+cos(phimcc))*Q_hat),
%
%  CALCULATE DIHEDRAL ROTATION ANGLE from rtarget to true mars target.
rot=atan2(dot(rmcc,cross(rtarget,rmars(:,2)))/rmcc_norm/rmag^2,
dot(rtarget,rmars(:,2))/rmag^2),
%  Rotate velocities to second part transfer trajectory (post-
midcourse)
i_hat=rtarget/norm(rtarget),
k_hat=rmcc/rmcc_norm,
j_hat=cross(k_hat,i_hat),
ROT=i_hat*(i_hat*cos(rot)-j_hat*sin(rot))'
+j_hat*(i_hat*sin(rot)+j_hat*cos(rot))' +k_hat*k_hat',
v2mcc=ROT*v1mcc;
v2=ROT*v;
%  CALCULATE MIDCOURSE CORRECTION VELOCITY CHANGE.
midcourse_deltav=v2mcc-v1mcc,
mccdeltavnorm=norm(midcourse_deltav)
%  FINALLY, DETERMINE END VELOCITY CONDITIONS.
%
vam_hyperbolicexcess=v2-vmars(:,2),
vam_he_norm=norm(vam_hyperbolicexcess),
vam_norm=sqrt(2*mum/(radiusm+100)+vam_he_norm^2)
%
end  ,

rearth =
          147292521.201715         -45354088.6904164
          26555828.5544097         -144812826.344405
                         0                         0
rmars =
          73523334.0725974         -246162443.428591
          215722719.460728          -11557574.074639
          2721384.48999595          5800844.23000556
vearth =
         -5.77105904654614          27.9417221210745
```

```
          29.2061674104939          -9.0132013599925
                         0                         0
vmars =
         -22.0060066348209          2.04909824339266
          9.87188464347986         -22.1371172128038
         0.747389877519146        -0.514797865778934
r0 =
          147292521.201715
          26555828.5544097
                         0
r =
         -246162443.428591
          -11557574.074639
          5800844.23000556
rmag =
          246501878.019227
deltaphi =
          3.00805425779427
mcc_hat =
       -0.0468993409237298
          0.99889962049293
                         0
rtarget =
           -246230632.4042
         -11560775.6155634
                         0
r =
           -246230632.4042
         -11560775.6155634
                         0
deltaphi1 =
          3.01013235744185
zmin =
     0
zmax =
          9.84460440108936
z_soln =
          8.15010967019599
fval =
     2.22044604925031e-016
exitflag =
     1
TOF =
         0.969972134632687
S =
         0.110541628111878
C =
         0.240385393531501
xp =
          2.03030261643546
yp =
         0.990899533175626
v0 =
         -7.40764367022531
          32.5000719466715
```

```
                           0
v =
          -1.4337644524506
         -20.3074208592907
                           0
vde_hyperbolicexcess =
         -1.63658462367917
          3.29390453617761
                           0
vde_he_norm =
          3.67807236523624
vde_norm =
          11.5284550276095
E =
         -331.170579177165
h =
                           0
                           0
          4983733651.56104
hnorm =
          4983733651.56104
h_hat =
     0
     0
     1
P =
          31362340650.9608
          13369753422.0407
                           0
Pnorm =
          34093206327.2594
P_hat =
         0.919900004414807
         0.392153008247595
                           0
Q_hat =
        -0.392153008247595
         0.919900004414807
                           0
e =
          0.25689007771112
p =
          187149698.329417
ans =
          145908338.957845
ans =
         -33332498.4771849
phi0 =
        -0.224594062134393
phimcc =
          1.21474196851256
rmcc_norm =
          171768446.831836
rmcc =
         -8055826.94790582
```

```
          171579436.352981
                         0
v1mcc =
         -29.2830397828643
          5.04402705612805
                         0
rot =
        0.0235348298943664
i_hat =
          -0.99889962049293
        -0.0468993409237298
                         0
k_hat =
        -0.0468993409237298
           0.99889962049293
                         0
j_hat =
                         0
                         0
                         1
ROT =
  Columns 1 through 2
         0.999723677797487    -1.29736050695634e-005
    -1.29736050695634e-005         0.999999390876205
       -0.0235067624898144       -0.00110366611960361
  Column 3
        0.0235067624898144
       0.00110366611960361
         0.999723068673692
midcourse_deltav =
        0.00802611483409521
       0.000376834156478267
         0.682782539387363
mccdeltavnorm =
         0.682829815265537
vam_hyperbolicexcess =
         -3.48220305443367
          1.82972732434008
          0.57091363860796
vam_he_norm =
           3.9748688753835
vam_norm =
          6.36717848711949
```

Now that the MATLAB scripts for calculating direct, retrograde, and Fimple transfer trajectories has been debugged, it is appropriate to put them into M-file form. In the following M-files, inputs include starting and ending planetary heliocentric positions, the starting heliocentric planetary velocity, and the elapsed time. Code calculates terminal v-infinity vectors and (for Fimple maneuver) the midcourse correction velocity change vector associated with the three intercept trajectories.

M-file code is as follows:,

```
function [v0,v]=
DirectIntercept(rstart,vstart,rfinish,deltat,mu);
%Calculates direct orbital transfer start -> finish in time
deltaut.
r0=rstart;r=rfinish;h=cross(rstart,vstart);
rmag=sqrt(dot(r,r));r0mag=sqrt(dot(r0,r0));
if dot(h,cross(r0,r))>0;
 deltaphi=acos(dot(r0,r)/(r0mag*rmag));
else
 deltaphi=2*pi-acos(dot(r0,r)/(r0mag*rmag));
end
B=sin(deltaphi)*sqrt(r0mag*rmag/(1-
cos(deltaphi)))/(r0mag+rmag);
TOF0=24*3600*(deltat)*sqrt(mu/(r0mag+rmag)^3);
TOF02=TOF0^2;
NTOFerr=@(z,B) Lambert(z(1),B(1))^2-TOF02;
z=zeros(1,1);
z=zeros(1,1);
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=2*z-1;
 end
end
zmin=z(1);
z=(2*pi)^2-.1;
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=z/2;
 end
end
zmax=2*z(1);
[z_soln,fval,exitflag]=fzero(@(z) Lambert(z,B(1)).^2-
TOF02,[zmin zmax]);
z=z_soln;
[TOF,S,C,xp,yp]=Lambert(z,B(1));
f=1-(r0mag+rmag)/r0mag*yp;
gdot=1-(r0mag+rmag)/rmag*yp;
g=B*sqrt(yp)*sqrt((r0mag+rmag)^3/mu);
```

```
v0=(r-f*r0)/g;
v=(gdot*r-r0)/g;
```

Test it:

```
mus=1.327151544E11;mu=mus;
rstart=rearth(:,1);rfinish=rmars(:,2);vstart=vearth(:,1);deltat=t_array
(2)-t_array(1);
[v0,v]= DirectIntercept(rstart,vstart,rfinish,deltat,mu)

v0 =
         -7.29240493942022
          31.9887517301508
          5.88562013387046
v =
         -1.52202981928112
          -19.998788814259
         -3.48582325762948
```

For the retrograde transfer we use the following M-file:

```
function [v0,v]=
RetrogradeIntercept(rstart,vstart,rfinish,deltat,mu);
%Calculates direct orbital transfer start -> finish in time
deltaut.
r0=rstart;r=rfinish;h=cross(rstart,vstart);
rmag=sqrt(dot(r,r));r0mag=sqrt(dot(r0,r0));
if dot(h,cross(r0,r))<0;
 deltaphi=acos(dot(r0,r)/(r0mag*rmag));
else
 deltaphi=2*pi-acos(dot(r0,r)/(r0mag*rmag));
end
B=sin(deltaphi)*sqrt(r0mag*rmag/(1-
cos(deltaphi)))/(r0mag+rmag);
TOF0=24*3600*(deltat)*sqrt(mu/(r0mag+rmag)^3);
TOF02=TOF0^2;
NTOFerr=@(z,B) Lambert(z(1),B(1))^2-TOF02;
z=zeros(1,1);
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=2*z-1;
 end
```

```
end
zmin=z(1);
z=(2*pi)^2-.1;
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=z/2;
 end
end
zmax=2*z(1);
[z_soln,fval,exitflag]=fzero(@(z) Lambert(z,B(1)).^2-
TOF02,[zmin zmax]);
z=z_soln;
[TOF,S,C,xp,yp]=Lambert(z,B(1));
f=1-(r0mag+rmag)/r0mag*yp;
gdot=1-(r0mag+rmag)/rmag*yp;
g=B*sqrt(yp)*sqrt((r0mag+rmag)^3/mu);
v0=(r-f*r0)/g;
v=(gdot*r-r0)/g;
```

Test it:

```
mus=1.327151544E11;mu=mus;
rstart=rearth(:,1);rfinish=rmars(:,2);vstart=vearth(:,1);deltat=t_array
(2)-t_array(1);
[v0,v]= RetrogradeIntercept(rstart,vstart,rfinish,deltat,mu)

v0 =
          1.58030762262941
        -32.7790027397511
        -5.84327627649743
v =
        -4.23188306971849
          19.585270462473
          3.59607817098754
```

For the Fimple transfer we have a more complicated M-file.

```
function [v0,deltavmcc,v]=
FimpleIntercept(rstart,vstart,rfinish,deltat,mu);
%Calculates Fimple's orbital transfer with midcourse plane-
change in time deltat.
r0=rstart;r=rfinish;h=cross(rstart,vstart);
rmag=sqrt(dot(r,r));r0mag=sqrt(dot(r0,r0));
if dot(h,cross(r0,r))>0;
```

```
 deltaphi=acos(dot(r0,r)/(r0mag*rmag));
else
 deltaphi=2*pi-acos(dot(r0,r)/(r0mag*rmag));
end

deltavmcc=zeros(3,1);
if deltaphi<pi/2;
%
B=sin(deltaphi)*sqrt(r0mag*rmag/(1-
cos(deltaphi)))/(r0mag+rmag);
TOF0=24*3600*(deltat)*sqrt(mu/(r0mag+rmag)^3);
TOF02=TOF0^2;
NTOFerr=@(z,B) Lambert(z(1),B(1))^2-TOF02;
z=zeros(1,1);
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=2*z-1;
 end
end
zmin=z(1);
z=(2*pi)^2-.1;
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=z/2;
 end
end
zmax=2*z(1);
[z_soln,fval,exitflag]=fzero(@(z) Lambert(z,B(1)).^2-
TOF02,[zmin zmax]);
z=z_soln;
[TOF,S,C,xp,yp]=Lambert(z,B(1));
f=1-(r0mag+rmag)/r0mag*yp;
gdot=1-(r0mag+rmag)/rmag*yp;
g=B*sqrt(yp)*sqrt((r0mag+rmag)^3/mu);
v0=(r-f*r0)/g;
v=(gdot*r-r0)/g;
%
%
else
%
```

```
%  Rotate destination about mcc direction into source orbit
plane
%   in order to form preliminary target
he=cross(rstart,vstart);he_hat=he/norm(he);
mcc_hat=cross(rfinish,he_hat);mcc_hat=mcc_hat/norm(mcc_hat)
;
rtarget=cross(he_hat,mcc_hat)*norm(rfinish);
r=rtarget;
% True anomaly difference to target has changed.  Need to
recalcualte.
if dot(h,cross(r0,r))>0,
 deltaphi1=acos(dot(r0,r)/(r0mag*rmag));
else
 deltaphi1=2*pi-acos(dot(r0,r)/(r0mag*rmag));
end
B=sin(deltaphi1)*sqrt(r0mag*rmag/(1-
cos(deltaphi1)))/(r0mag+rmag);
TOF0=24*3600*(deltat)*sqrt(mu/(r0mag+rmag)^3);
TOF02=TOF0^2;
NTOFerr=@(z,B) Lambert(z(1),B(1))^2-TOF02;
z=zeros(1,1);
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=2*z-1;
 end
end
zmin=z(1);
z=(2*pi)^2-.1;
for jcount=1:100;
 result=Lambert(z,B(1)).^2-TOF02;
 if result<0;
  break
 else
  z=z/2;
 end
end
zmax=2*z(1);
[z_soln,fval,exitflag]=fzero(@(z) Lambert(z,B(1)).^2-
TOF02,[zmin zmax]);
z=z_soln(1);
[TOF,S,C,xp,yp]=Lambert(z,B(1));
f=1-(r0mag+rmag)/r0mag*yp(1);
gdot=1-(r0mag+rmag)/rmag*yp(1);
```

```
g=B(1)*sqrt(yp(1))*sqrt((r0mag+rmag)^3/mu);
v0=(r-f*r0)/g;
v=(gdot*r-r0)/g;
%
%  FIND Fimple's MIDCOURSE CORRECTION POINT 90 degrees
before rtarget.
%  -Find parameters of first part of transfer trajectory
(pre-midcourse)
E=dot(v0,v0)/2-mu/r0mag;
h=cross(r0,v0);hnorm=sqrt(dot(h,h));h_hat=h/hnorm;
P=-(cross(h,v0)+r0*mu/r0mag);
Pnorm=sqrt(dot(P,P));
P_hat=P/Pnorm;
Q_hat=cross(h_hat,P_hat);
e=Pnorm/mu;
p=dot(h,h)/mu;
dot(r0,P_hat);
dot(r0,Q_hat);
phi0=atan2(dot(r0,Q_hat),dot(r0,P_hat));
%  Find mid-course correction (mc) location and pre-burn
velocity.
phimcc=phi0+deltaphi1-pi/2.;
rmcc_norm=p/(1+e*cos(phimcc));
rmcc=rmcc_norm*(P_hat*cos(phimcc)+Q_hat*sin(phimcc));
v1mcc=sqrt(mu/p)*(-
sin(phimcc)*P_hat+(e+cos(phimcc))*Q_hat);
%
%  CALCULATE DIHEDRAL ROTATION ANGLE from rtarget to true
mars target.
rot=atan2(dot(rmcc,cross(rtarget,rfinish))/rmcc_norm/rmag^2
, dot(rtarget,rfinish)/rmag^2);
%  Rotate velocities to second part transfer trajectory
(post-midcourse)
i_hat=rtarget/norm(rtarget);
k_hat=rmcc/rmcc_norm;
j_hat=cross(k_hat,i_hat);
ROT=i_hat*(i_hat*cos(rot)-j_hat*sin(rot))'
+j_hat*(i_hat*sin(rot)+j_hat*cos(rot))' +k_hat*k_hat';
v2mcc=ROT*v1mcc;
v=ROT*v;
%  CALCULATE MIDCOURSE CORRECTION VELOCITY CHANGE.
deltavmcc=v2mcc-v1mcc;
%
end
```

Then, test this M-file.

```
mus=1.327151544E11;mu=mus;
rstart=rearth(:,1);rfinish=rmars(:,2);vstart=vearth(:,1);deltat=t_array
(2)-t_array(1);
[v0,deltavmcc,v]= FimpleIntercept(rstart,vstart,rfinish,deltat,mu)

v0 =
         -7.40764367022531
          32.5000719466715
                         0
deltavmcc =
       0.00802611483409521
      0.000376834156480932
         0.682782539387363
v =
         -1.43310481104094
         -20.3073898884637
         0.0561157728290243
```

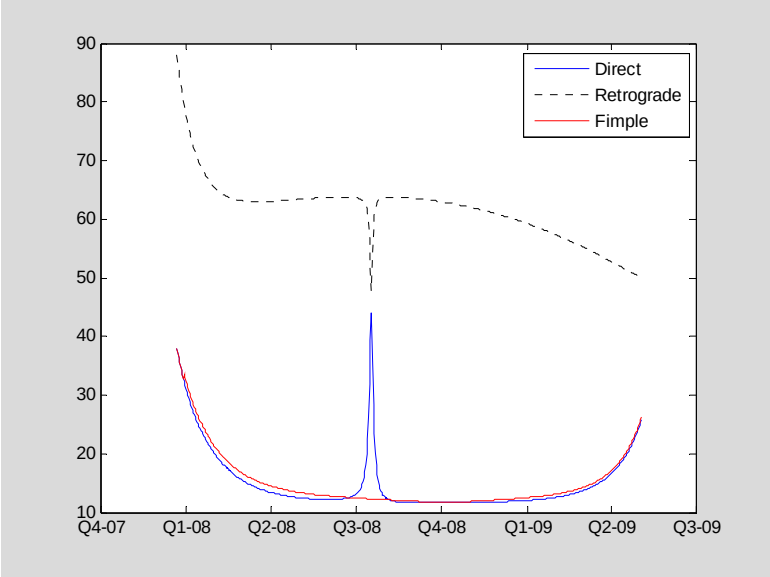
.

It is not clear *a priori* which of the three transfer options, i.e., Fimple, direct, or retrograde, is optimal in terms of requiring the least total velocity change be supplied by rocket systems. All should be calculated and the best chosen for each pair of departure and arrival times. At planetary arrival or departure for each trajectory we subtract the appropriate helioceentric planetary velocity vector in order to calculate the hyperbolic excess velocity. Then we calculate the speed at ground level with respect to the planet. For the Fimple transfer in which orbital plane change occurs away from the planets, we also need to calculate the associated magnitude of that velocity change. For each of these 2 or 3 trajectories in each transfer direction, we thus sum the calculated Δv magnitudes, then for each transfer direction we choose the one with the least total magnitude. A variation in the procedure is whether or not we include in the total Δv the deceleration magnitudes at the arrivl planet, since those could alternatively be accomplished by either retrorocket firing or aerodynamic friction during atmospheric reentry.

To calculate the total change in velocity to go from earth's *location* to mars' *location* using the martian atmosphere to slow down upon arrival, we have the following code. Note that this code fails to consider the change in travel times caused by planetary acceleration at the two ends of the transfer. We go into this code with start and finish times initially identified.

```
tstart=datenum('1-November-2007');
deltavD=zeros(500,1);
deltavR=deltavD;
deltavF=deltavD;
tfinish=deltavR;
```

```
for i=1:500;
tfinish(i)=tstart+50+i;
mus=1.327151544E11;
mue=3.986e5;
mum=4.305e4;
mu=mus;
radiuse=6378;radiusm=3380;
[rearth,rmars,vearth,vmars]=EarthMars([tstart tfinish(i)]);
rstart=rearth(:,1);
rfinish=rmars(:,2);
vstart=vearth(:,1);
deltat=tfinish(i)-tstart;
[v0D,vD]= DirectIntercept(rstart,vstart,rfinish,deltat,mu);
vdeheD=v0D-vearth(:,1);
vdeheD_norm=norm(vdeheD);
vdeD_norm=sqrt(2*mue/(radiuse)+vdeheD_norm^2);
[v0R,vR]= RetrogradeIntercept(rstart,vstart,rfinish,deltat,mu);
vdeheR=v0R-vearth(:,1);
vdeheR_norm=norm(vdeheR);
vdeR_norm=sqrt(2*mue/(radiuse)+vdeheR_norm^2);
[v0F,deltavmcc,vF]= FimpleIntercept(rstart,vstart,rfinish,deltat,mu);
vdeheF=v0F-vearth(:,1);
vdeheF_norm=norm(vdeheF);
vdeF_norm=sqrt(2*mue/(radiuse)+vdeheF_norm^2);
deltavD(i)=vdeD_norm;
deltavR(i)=vdeR_norm;
deltavF(i)=vdeF_norm+norm(deltavmcc);
end
plot(tfinish,deltavD,'b',tfinish,deltavR,'k:',tfinish,deltavF,'r'),date
tick('x'),
legend('Direct','Retrograde','Fimple','location','NorthEast')
```



The required total velocity change magnitude for direct and retrograde transfers changes quickly with duration for transfers using about half of an orbit. This is due to the small

inclination of the orbital plane of mars with respect to earth's orbtal plane. The Fimple transfer avoids this issue and is obviously optimal for such transfers.

The above computational procedure is not entirely complete. The problem with it is that motion through the regions near the two planets occurs more rapidly than the heliocentric Keplerian trajectory predicts, due to the gravitational attractions of the two planets. This puts the entire computed trajectory result in error by a period of tune which is perhaps as much as several days. Since the planets move during this time, the computed trajectory is also in error spatially. We could either choose to ignore this error or try to reduce it.

To do a more accurate job it is appropriate to use the "patched conic" approximation in which the gravitational fields of the Earth, Sun, and Mars are "turned on and off" in different spatial regions. Formulae for the size of the "sphere of influence" surrounding each planet in which this approximation "switches on" the planet's gravity field were derived by Laplace based on minimizing the resulting error. (His recommended SOI radius is the 2/5 power of the ratio of planet to solar masses multiplied by the planet's distance from the sun.) Although this switching obviously does not happen in reality, according to several reference books the "patched conic" approximation has been found to yield sufficiently accurate results for many mission planning activities while allowing the simple formulas associated with Keplerian orbits to be used. The "patched-conic" approximation also can form a good starting point for iterative calculation of a precise final trajectory, using codes which numerically integrate forces from all non-negligible sources.

In the "patched conic" interplanetary approximation we divide the transfer trajectory into three consecutive parts with only a single central gravitational mass acting in each part. Within the "sphere of influence" region surrounding each planet, a planet-centered hyperbolic trajectory is found connecting the planet's surface to the "sphere of influence" boundary, where it meets the heliocentric trajectory. An iterative procedure is necessary to patch the three conics together so that they connect smoothly in position and velocity at sphere-of-influence-crossing boundary times. The heliocentric transfer trajectory calculated between planet center locations and times should form the starting point for this iterative procedure.

None of the reference books surveyed by the author included an explicit computational algorithm for the patched conic interplanetary transfer trajectory, although several discussed it in general terms. A proposed iterative algorithm was therefore devised by the author, as follows.

---

(1) Calculate heliocentric trajectory transitioning between the two planets' locations at the specified end dates (using LambertGauss-style algorithms), neglecting the planetary gravitational field effects. Determine the trajectory's heliocentric velocity vectors at planet center locations and end times.

---

(2) Subtract planet velocity vectors to determine hyperbolic excess velocities.
(3) Use conservation of energy to find corresponding horizontal speed in planet's upper atmosphere or surface, wherever horizontal motion is being required.
(4) Calculate hyperbolic time-of-flight between planet's surface and boundary of SOI (sphere of influence).
(5) Select patch position on SOI boundary relative to planet yielding an acceptable terminal location at the planet.
(6) Allowing time for that hyperbolic trajectory to occur, find the planet's position at the time shifted by the time interval calculated in (4) from the specified endpoint time.
(7) Calculate the heliocentric location of the patch position on the SOI boundary which was determined relative to the planet in (5).;

(8) Calculate the distance (error) between the heliocentric trajectory positions at the shifted times calculated in (6) and the patch positions determined in (6).
(9)If both terminal distance errors calculated in (8) are sufficiently small, stop iterating and exit the procedure.
(10) Otherwise, calculate a new Lambert trajectory connecting the two heliocentric patch points calculated in (6) for the shifted times calculated in (5). Determine the new trajectory's heliocentric velocities at its specified end points and times (which were the patched end points and shifted times from the previous iteration).
(11) Go back to step (2).

It seems likely that this procedure should converge quickly. Shifts in patch positions calculated in step (5) of successive iterations will be small compared to the overall transfer trajectory. End-point velocities are not very sensitive to small changes in end-point positions. The resulting changes in end-point velocities will therefore be smaller still as will resulting changes in calculated speeds near planets and hyperbolic time-of-flight values in steps (3) and (4).

It is known that neither terminating planet-centered trajectory could be elliptic or parabolic. Both are necessarily hyperbolic. For hyperbolic trajectories we can use the following:

$$v_p = \sqrt{v_\infty^2 + \frac{2\mu}{r_p}} \tag{B-31}$$

$$h = r_p v_p \tag{B-32}$$

$$P = \sqrt{\mu^2 + h^2 v_\infty^2} \tag{B-33}$$

$$e = \frac{P}{\mu} \tag{B-34}$$

$$r(t) = \frac{\mu}{v_\infty^2}\left(-1 + e\cosh\left(H(t)\right)\right) \tag{B-35}$$

$$\left(t - t_p\right)\frac{v_\infty^3}{\mu} = e\sinh\left(H(t)\right) - H(t) \tag{B-36}$$

Equations (31)-(36) can be solved for the time needed to transit between the SOI boundary and periapsis:

$$\begin{aligned} H\left(t - t_p\right) &= \cosh^{-1}\left(\frac{\mu + v_\infty^2 r_{SOI}}{\mu + v_\infty^2 r_p}\right) \\ &\equiv \ln\left(\frac{\mu + v_\infty^2 r_{SOI}}{\mu + v_\infty^2 r_p} + \sqrt{\left(\frac{\mu + v_\infty^2 r_{SOI}}{\mu + v_\infty^2 r_p}\right)^2 - 1}\right) \end{aligned} \tag{B-37}$$

$$\boxed{\begin{aligned} \left(t - t_p\right) &= \frac{\mu + v_\infty^2 r_p}{v_\infty^3}\sinh\left(\cosh^{-1}\left(\frac{\mu + v_\infty^2 r_{SOI}}{\mu + v_\infty^2 r_p}\right)\right) - \frac{\mu}{v_\infty^3}\cosh^{-1}\left(\frac{\mu + v_\infty^2 r_{SOI}}{\mu + v_\infty^2 r_p}\right) \\ &\equiv \frac{\mu + v_\infty^2 r_p}{v_\infty^3}\sqrt{\left(\frac{\mu + v_\infty^2 r_{SOI}}{\mu + v_\infty^2 r_p}\right)^2 - 1} \quad - \frac{\mu}{v_\infty^3}\ln\left(\frac{\mu + v_\infty^2 r_{SOI}}{\mu + v_\infty^2 r_p} + \sqrt{\left(\frac{\mu + v_\infty^2 r_{SOI}}{\mu + v_\infty^2 r_p}\right)^2 - 1}\right) \end{aligned}} \tag{B-38}$$

Either equivalent form of Equation (B-38) may be used to evaluate step (4) of the above algorithm.

Computed trajectories should be tangent to the planet's surface (or perhaps to a specified altitude above the surface) so that departures can leave after spending some time in a low "parking orbit" around the pplanet and so that arrivals enter the target planet's atmosphere with an appropriate re-entry angle. This constrains the "impact parameter", i.e., the asymptotic offset distance from a trejectory aligned with the planet's center, to a particular value given by the following function of periapsis distance and hyperbolic excess speed.

$$b = \sqrt{r_p^2 + \frac{2\mu r_p}{v_\infty^2}} \tag{B-39}$$

For step (5) of the algorithm, it is also necessary to specify how a suitable terminal location with respect to the planet is to be selected. The locus of possible terminal

locations forms a circle on the planet's surface, but not a great circle. The locus of possible terminal locations forms circles on planet surfaces which are an angle (η) from the planetocentric asymptotic direction of motion on the arrival planet and (π–η) from the asymptotic direction of motion on the departure planet. The circle's size angle is based on the hyperbolic asympote's true anomaly with respect to periapsis, which is given by the following formula:

$$\cos\eta = -\frac{1}{e} = -\frac{\mu}{\mu + v_\infty^2 r_p} \qquad \text{.(B-40)}$$

$$\frac{\pi}{2} \le \eta < \pi$$

The choice of a terminal location considers the desire to land on or near the equator during daylight, although that is not possible in every case. The criteria used herein is as follows.
(a) If the circle of possible terminal locations is parallel to but distinct from the equator, the point chosen on the terminal circle is at local noon with respect to the sun.
(b) If the circle of possible terminal locations intersects the equator, the point chosen is the equator-crossing point closest to local noon.
(c) Otherwise the point chosen is the closest approach to the equator.

The algorithm implementing this must know the spatial orientation of the "north" direction for each planet in the heliocentric coordinate system. These are known in the earth-oriented system using right ascension and declination angles, and will be converted into heliocentric longtitude and latitude.

A recent paper found on-line by [Duxbury, et al], researchers from JPL, NASA centers, and the US Geological Survey, include the following description of Mars' spin axis orientation:

> The right ascension α and declination δ in degrees at a given time *t* are given by the expressions:
>
> α = 317.68143º - 0.1061º/century * T
>
> δ = 52.88650º - 0.0609º/century * T
>
> where T is the number of Julian centuries of *t* from the standard epoch of J2000.0.

Ignoring the small drift rates we thus have the following descriptions of the respective northbdirections.

	Right Ascension (degrees)	Declination (degrees)
Earth's North Direction	0 (or undefined)	90.00000
Mars' North Direction	317.68143	52.88650

Spatial directions can be converted from the (Right Ascension=α, Declination=δ) descriptive system to the (Ecliptic Longtitude=λ,Ecliptic Latitude=β) descriptive system through the following formulae [Duffett-Smith 1988, 42]:

$$\beta = \sin^{-1}\left(\sin\delta\cos\varepsilon - \cos\delta\sin\varepsilon\sin\alpha\right)$$

$$x = \cos\alpha$$

$$y = \sin\alpha\cos\varepsilon + \tan\delta\sin\varepsilon$$

$$\lambda = \tan^{-1}\left(\frac{y}{x}\right)$$

where ε is the "obliquity of the ecliptic", ε=23.441884 degrees.

The conversion script follows:

```
epsilon=23.441884*pi/180;
%
alpham=317.68143*pi/180;deltam=52.88650*pi/180;
betam=asin(sin(deltam)*cos(epsilon)-
cos(deltam)*sin(epsilon)*sin(alpham));
lambdam=atan2(sin(alpham)*cos(epsilon)+tan(deltam)*sin(epsilon),cos(alp
ham));
betam_deg=betam*180/pi,lambdam_deg=lambdam*180/pi,
nm=[cos(lambdam)*cos(betam) sin(lambdam)*cos(betam) sin(betam)]'
%
alphae=0;deltae=pi/2;
betae=asin(sin(deltae)*cos(epsilon)-
cos(deltae)*sin(epsilon)*sin(alphae));
lambdae=atan2(sin(alphae)*cos(epsilon)+tan(deltae)*sin(epsilon),cos(alp
hae));
betae_deg=betae*180/pi,lambdae_deg=lambdae*180/pi,
ne=[cos(lambdae)*cos(betae) sin(lambdae)*cos(betae) sin(betae)]'

betam_deg =
          63.2823689927763
lambdam_deg =
          -7.0872490208687
nm =
         0.446158726935355
        -0.0554711783756754
         0.893233081983105
betae_deg =
                  66.558116
lambdae_deg =
    90
ne =
     2.43593683899589e-017
          0.39781867566908
          0.917464059943984
```

(0) Start with a known unit vector, $\hat{n}$, pointing in the "north pole" direction of a planet. (These are provided by the script above.)
(1) In planetocentric coordinates, determine the following unit vector pointing away from the planet in the asymptotic direction of the heliocentric transfer trajectory:

$$\hat{w} = \begin{cases} +\dfrac{\vec{\mathrm{v}}_\infty}{\mathrm{v}_\infty} & \text{for departure} \\ -\dfrac{\vec{\mathrm{v}}_\infty}{\mathrm{v}_\infty} & \text{for arrival} \end{cases}$$

(2) Determine the unit vector pointing torwards the sun from the heliocentric planet location vector, as

$$\hat{s} = -\frac{\vec{r}_{\text{planet}}}{r_{\text{planet}}}$$

(3) Calculate η, the angle from $\hat{w}$ to the circle of possible terminal locations, via

$$\eta = \cos^{-1}\left(-\frac{\mu}{\mu + v_\infty^2 r_p} \right)$$

(4) If $\hat{n} \times \hat{w} = \vec{0}$ then the terminal circle is parallel to the planets's equator. In planetocentric coordinates we select the terminal location as follows:

$$\vec{r}_t = r_p \left(\hat{w}\cos\eta + \frac{\hat{s} - (\hat{s} \bullet \hat{w})\hat{w}}{\left\| \hat{s} - (\hat{s} \bullet \hat{w})\hat{w} \right\|} \sin\eta \right)$$

(5) Else if $\left| \hat{n} \bullet \hat{w} \right| < \sin\eta$ then the terminal circle crosses the planet's equator. The target location is then

$$\vec{r}_t = r_p \begin{cases} +\dfrac{\hat{n} \times \hat{w}}{\left\| \hat{n} \times \hat{w} \right\|} \sin\eta - \dfrac{\hat{n} \times (\hat{n} \times \hat{w})}{\left\| \hat{n} \times (\hat{n} \times \hat{w}) \right\|} \cos\eta & \text{if } \hat{\mathrm{s}} \bullet (\hat{n} \times \hat{w}) > 0 \\ -\dfrac{\hat{n} \times \hat{w}}{\left\| \hat{n} \times \hat{w} \right\|} \sin\eta - \dfrac{\hat{n} \times (\hat{n} \times \hat{w})}{\left\| \hat{n} \times (\hat{n} \times \hat{w}) \right\|} \cos\eta & \text{if } \hat{\mathrm{s}} \bullet (\hat{n} \times \hat{w}) \le 0 \end{cases}$$

(6) Else, the terminal circle approaches but misses the planet's equator. The terminal location closest to the planet's equator is then:

$$\vec{r}_t = r_p \begin{cases} +\hat{w}\cos\eta + \dfrac{\hat{n} - (\hat{n} \bullet \hat{w})\hat{w}}{\left\| \hat{n} - (\hat{n} \bullet \hat{w})\hat{w} \right\|} \sin\eta & \text{if } (\hat{n} \bullet \hat{w}) > 0 \\ +\hat{w}\cos\eta - \dfrac{\hat{n} - (\hat{n} \bullet \hat{w})\hat{w}}{\left\| \hat{n} - (\hat{n} \bullet \hat{w})\hat{w} \right\|} \sin\eta & \text{if } (\hat{n} \bullet \hat{w}) \le 0 \end{cases}$$

After the target location has been identified by one of the branches of this algorithm, we calculate

$$\hat{b} = \frac{\vec{r}_t - (\vec{r}_t \bullet \hat{w})\hat{w}}{\left\| \vec{r}_t - (\vec{r}_t \bullet \hat{w})\hat{w} \right\|}$$

and

$$\vec{b} = \hat{b}\sqrt{r_p^2 + \frac{2\mu r_p}{v_\infty^2}}$$

Then, the patch location is

$$\vec{r}_{\text{patch}} = \sqrt{r_{SOI}{}^2 - b^2}\,\hat{w} + \vec{b}$$

MATLAB script implementing this algorithm follows.

```
% Setup parameters
tstart=datenum('1-November-2007');
tfinish=datenum('15-July-2008');
mus=1.327151544E11;
mue=3.986e5;
mum=4.305e4;
mu=mus;
radiuse=6378;radiusm=3380;
% Calculate heliocentric transfer trajectories between planet centers.
[rearth,rmars,vearth,vmars]=EarthMars([tstart tfinish]);
rstart=rearth(:,1);
rfinish=rmars(:,2);
vstart=vearth(:,1);
deltat=tfinish-tstart;
%  Laplace's recommended "Sphere-Of-Influence" sizes
RSOIe=norm(rearth(:,1))*(mue/mus)^0.4;
RSOIm=norm(rmars(:,2))*(mum/mus)^0.4;
%
[v0D,vD]= DirectIntercept(rstart,vstart,rfinish,deltat,mu);
vdeheD=v0D-vearth(:,1);
vdeheD_norm=norm(vdeheD);
vdeD_norm=sqrt(2*mue/(radiuse)+vdeheD_norm^2);
vamheD=vD-vmars(:,2);
vamheD_norm=norm(vamheD);
vamD_norm=sqrt(2*mum/(radiusm)+vamheD_norm^2);
%
[v0R,vR]= RetrogradeIntercept(rstart,vstart,rfinish,deltat,mu);
vdeheR=v0R-vearth(:,1);
vdeheR_norm=norm(vdeheR);
vdeR_norm=sqrt(2*mue/(radiuse)+vdeheR_norm^2);
vamheR=vR-vmars(:,2);
vamheR_norm=norm(vamheR);
vamR_norm=sqrt(2*mum/(radiusm)+vamheR_norm^2);
%
[v0F,deltavmcc,vF]= FimpleIntercept(rstart,vstart,rfinish,deltat,mu);
vdeheF=v0F-vearth(:,1);
vdeheF_norm=norm(vdeheF);
vdeF_norm=sqrt(2*mue/(radiuse)+vdeheF_norm^2);
vamheF=vF-vmars(:,2);
vamheF_norm=norm(vamheF);
vamF_norm=sqrt(2*mum/(radiusm)+vamheF_norm^2);
%
```

```
deltavD=vdeD_norm,
deltavR=vdeR_norm,
deltavF=vdeF_norm+norm(deltavmcc),
if deltavF<=min(deltavD,deltavR);
 mode='F';
 vdehe=vdeheF;vdehe_norm=vdeheF_norm;vde_norm=vdeF_norm;
 vamhe=vamheF;vamhe_norm=vamheF_norm;vam_norm=vamF_norm;
elseif deltavD<min(deltavF,deltavR);
 mode='D';
 vdehe=vdeheD;vdehe_norm=vdeheD_norm;vde_norm=vdeD_norm;
 vamhe=vamheD;vamhe_norm=vamheD_norm;vam_norm=vamD_norm;
else;
 mode='R';
 vdehe=vdeheR;vdehe_norm=vdeheR_norm;vde_norm=vdeR_norm;
 vamhe=vamheR;vamhe_norm=vamheR_norm;vam_norm=vamR_norm;
end
%
% The "patched conic" loop follows:
%
for kcount=1:5;  %Don't execute this more than ten times.
% Determine patch points in planetocentric coordinates.
 etae=acos(mue/(mue+radiuse*vdehe_norm^2));
 we=+vdehe/vdehe_norm;
 se=-rearth(:,1)/norm(rearth(:,1));
 if (norm(cross(ne,we))==0);
  rte=radiuse*(we*cos(etae)+sin(etae)*(se-dot(se,we)*we)/norm(se-
dot(se,we)*we));
 elseif abs(dot(ne,we))<sin(etae);
  if dot(se,cross(ne,we))>0;
   rte=radiuse*(+cross(ne,we)/norm(cross(ne,we))*sin(etae)-
cross(ne,cross(ne,we))/norm(cross(ne,cross(ne,we)))*cos(etae));
  else
   rte=radiuse*(-cross(ne,we)/norm(cross(ne,we))*sin(etae)-
cross(ne,cross(ne,we))/norm(cross(ne,cross(ne,we)))*cos(etae));
  end
 else
  if dot(ne,we)>0;
rte=radiuse*(we*cos(etae)+sin(etae)*cross(we,cross(ne,we))/norm(cross(w
e,cross(ne,we))));
  else
rte=radiuse*(we*cos(etae)-
sin(etae)*cross(we,cross(ne,we))/norm(cross(we ,cross(ne,we))));
  end
end
be_hat=cross(we,cross(rte,we))/norm(cross(we,cross(rte,we)));
be_norm=sqrt(radiuse^2+2*mue*radiuse/vdehe_norm^2);
be=be_hat*be_norm;
%
 etam=acos(mum/(mum+radiusm*vamhe_norm^2));
 wm=-vamhe/vamhe_norm;
 sm=-rmars(:,2)/norm(rmars(:,2));
 if (norm(cross(nm,wm))==0)
  rtm=radiusm*(wm*cos(etam)+sin(etam)*(sm-dot(sm,wm)*wm)/norm(sm-
dot(sm,wm)*wm));
 elseif abs(dot(nm,wm))<sin(etam);
```

```
  if dot(sm,cross(nm,wm))>0;
   rtm=radiusm*(+cross(nm,wm)/norm(cross(nm,wm))*sin(etam)-
cross(nm,cross(nm,wm))/norm(cross(nm,cross(nm,wm)))*cos(etam));
  else
   rtm=radiusm*(-cross(nm,wm)/norm(cross(nm,wm))*sin(etam)-
cross(nm,cross(nm,wm))/norm(cross(nm,cross(nm,wm)))*cos(etam));
  end
 else
  if dot(nm,wm)>0;
rtm=radiusm*(wm*cos(etam)+sin(etam)*cross(wm,cross(nm,wm))/norm(cross(w
m,cross(nm,wm))));
  else
rtm=radiusm*(wm*cos(etam)-
sin(etam)*cross(wm,cross(nm,wm))/norm(cross(wm ,cross(nm,wm))));
  end
end
bm_hat=cross(wm,cross(rtm,wm))/norm(cross(wm,cross(rtm,wm)));
bm_norm=sqrt(radiusm^2+2*mum*radiusm/vamhe_norm^2);
bm=bm_hat*bm_norm;
%
 He=acosh( (mue+RSOIe*vdehe_norm^2)/(mue+radiuse*vdehe_norm^2));
 deltate=((mue+radiuse*vdehe_norm^2)*sinh(He)-mue*He)/vdehe_norm^3
/24/3600;
 Hm=acosh( (mum+RSOIm*vamhe_norm^2)/(mum+radiusm*vamhe_norm^2));
 deltatm=((mum+radiusm*vamhe_norm^2)*sinh(Hm)-mum*Hm)/vamhe_norm^3
/24/3600;
tpatche=tstart+deltate;
tpatchm=tfinish-deltatm;
[rearth,rmars,vearth,vmars]=EarthMars([tstart tfinish tpatche
tpatchm]);
rpatche=rearth(:,3)+be+we*sqrt(RSOIe^2-be_norm^2);
rpatchm=rmars(:,4)+bm+wm*sqrt(RSOIm^2-bm_norm^2);
if mode=='D';
[v0,v]= DirectIntercept(rpatche,vearth(:,3),rpatchm,tpatchm-
tpatche,mu);deltavmcc=zeros(3,1);
elseif mode=='R';
[v0,v]= RetrogradeIntercept(rpatche,vearth(:,3),rpatchm,tpatchm-
tpatche,mu);deltavmcc=zeros(3,1);
else
[v0,deltavmcc,v]= FimpleIntercept(rpatche,vearth(:,3),rpatchm,tpatchm-
tpatche,mu);
end
vdehe=v0-vearth(:,3);vdehe_norm=norm(vdehe);
vde_norm=sqrt(2*mue/(radiuse)+vdehe_norm^2);
vamhe=v-vmars(:,4);
vamhe_norm=norm(vamhe);
vam_norm=sqrt(2*mum/(radiusm)+vamhe_norm^2);
%
deltav=vde_norm+norm(deltavmcc)
%
end

deltavD =
          26.2210261687241
deltavR =
```

```
          59.4794383184961
deltavF =
          12.3146777910176
deltav =
          12.3156012651619
deltav =
          12.3158664161029
deltav =
          12.3158659985442
deltav =
          12.3158662179394
deltav =
           12.315866212284
```

Next, we adapt this script to scanning through boundary time definitions.

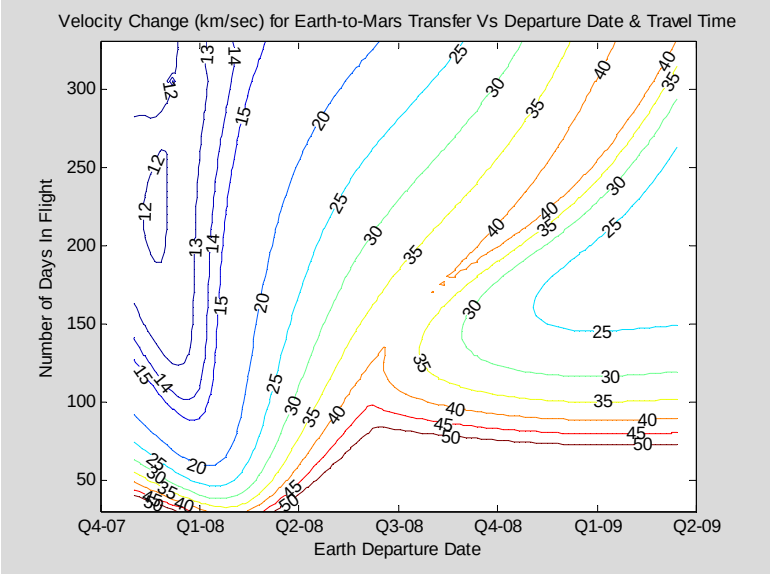
```
% Setup parameters
mus=1.327151544E11;
mue=3.986e5;
mum=4.305e4;
mu=mus;
radiuse=6378;radiusm=3380;
tstart0=datenum('1-November-2007');
tearth=tstart0+(0:5:500);
ttransfer=30:5:330;
DeltaVRocket=zeros(length(tearth),length(ttransfer));
for istart=1:length(tearth);
for jtransfer=1:length(ttransfer);
tstart=tearth(istart);tfinish=tstart+ttransfer(jtransfer);
% Calculate heliocentric transfer trajectories between planet centers.
[rearth,rmars,vearth,vmars]=EarthMars([tstart tfinish]);
rstart=rearth(:,1);
rfinish=rmars(:,2);
vstart=vearth(:,1);
deltat=tfinish-tstart;
%  Laplace's recommended "Sphere-Of-Influence" sizes
RSOIe=norm(rearth(:,1))*(mue/mus)^0.4;
RSOIm=norm(rmars(:,2))*(mum/mus)^0.4;
%
[v0D,vD]= DirectIntercept(rstart,vstart,rfinish,deltat,mu);
vdeheD=v0D-vearth(:,1);
vdeheD_norm=norm(vdeheD);
vdeD_norm=sqrt(2*mue/(radiuse)+vdeheD_norm^2);
vamheD=vD-vmars(:,2);
vamheD_norm=norm(vamheD);
vamD_norm=sqrt(2*mum/(radiusm)+vamheD_norm^2);
%
[v0R,vR]= RetrogradeIntercept(rstart,vstart,rfinish,deltat,mu);
vdeheR=v0R-vearth(:,1);
vdeheR_norm=norm(vdeheR);
vdeR_norm=sqrt(2*mue/(radiuse)+vdeheR_norm^2);
vamheR=vR-vmars(:,2);
vamheR_norm=norm(vamheR);
vamR_norm=sqrt(2*mum/(radiusm)+vamheR_norm^2);
```

```
%
[v0F,deltavmcc,vF]= FimpleIntercept(rstart,vstart,rfinish,deltat,mu);
vdeheF=v0F-vearth(:,1);
vdeheF_norm=norm(vdeheF);
vdeF_norm=sqrt(2*mue/(radiuse)+vdeheF_norm^2);
vamheF=vF-vmars(:,2);
vamheF_norm=norm(vamheF);
vamF_norm=sqrt(2*mum/(radiusm)+vamheF_norm^2);
%
deltavD=vdeD_norm,
deltavR=vdeR_norm,
deltavF=vdeF_norm+norm(deltavmcc),
if deltavF<=min(deltavD,deltavR);
 mode='F';
 vdehe=vdeheF;vdehe_norm=vdeheF_norm;vde_norm=vdeF_norm;
 vamhe=vamheF;vamhe_norm=vamheF_norm;vam_norm=vamF_norm;
elseif deltavD<min(deltavF,deltavR);
 mode='D';
 vdehe=vdeheD;vdehe_norm=vdeheD_norm;vde_norm=vdeD_norm;
 vamhe=vamheD;vamhe_norm=vamheD_norm;vam_norm=vamD_norm;
else;
 mode='R';
 vdehe=vdeheR;vdehe_norm=vdeheR_norm;vde_norm=vdeR_norm;
 vamhe=vamheR;vamhe_norm=vamheR_norm;vam_norm=vamR_norm;
end
%
% The "patched conic" loop follows:
%
for kcount=1:5;  %Don't execute this more than ten times.
% Determine patch points in planetocentric coordinates.
 etae=acos(mue/(mue+radiuse*vdehe_norm^2));
 we=+vdehe/vdehe_norm;
 se=-rearth(:,1)/norm(rearth(:,1));
 if (norm(cross(ne,we))==0);
  rte=radiuse*(we*cos(etae)+sin(etae)*(se-dot(se,we)*we)/norm(se-
dot(se,we)*we));
 elseif abs(dot(ne,we))<sin(etae);
  if dot(se,cross(ne,we))>0;
   rte=radiuse*(+cross(ne,we)/norm(cross(ne,we))*sin(etae)-
cross(ne,cross(ne,we))/norm(cross(ne,cross(ne,we)))*cos(etae));
  else
   rte=radiuse*(-cross(ne,we)/norm(cross(ne,we))*sin(etae)-
cross(ne,cross(ne,we))/norm(cross(ne,cross(ne,we)))*cos(etae));
  end
 else
  if dot(ne,we)>0;
rte=radiuse*(we*cos(etae)+sin(etae)*cross(we,cross(ne,we))/norm(cross(w
e,cross(ne,we))));
  else
rte=radiuse*(we*cos(etae)-
sin(etae)*cross(we,cross(ne,we))/norm(cross(we ,cross(ne,we))));
  end
end
be_hat=cross(we,cross(rte,we))/norm(cross(we,cross(rte,we)));
be_norm=sqrt(radiuse^2+2*mue*radiuse/vdehe_norm^2);
```

```
be=be_hat*be_norm;
%
 etam=acos(mum/(mum+radiusm*vamhe_norm^2));
 wm=-vamhe/vamhe_norm;
 sm=-rmars(:,2)/norm(rmars(:,2));
 if (norm(cross(nm,wm))==0)
  rtm=radiusm*(wm*cos(etam)+sin(etam)*(sm-dot(sm,wm)*wm)/norm(sm-
dot(sm,wm)*wm));
 elseif abs(dot(nm,wm))<sin(etam);
  if dot(sm,cross(nm,wm))>0;
   rtm=radiusm*(+cross(nm,wm)/norm(cross(nm,wm))*sin(etam)-
cross(nm,cross(nm,wm))/norm(cross(nm,cross(nm,wm)))*cos(etam));
  else
   rtm=radiusm*(-cross(nm,wm)/norm(cross(nm,wm))*sin(etam)-
cross(nm,cross(nm,wm))/norm(cross(nm,cross(nm,wm)))*cos(etam));
  end
 else
  if dot(nm,wm)>0;
rtm=radiusm*(wm*cos(etam)+sin(etam)*cross(wm,cross(nm,wm))/norm(cross(w
m,cross(nm,wm))));
  else
rtm=radiusm*(wm*cos(etam)-
sin(etam)*cross(wm,cross(nm,wm))/norm(cross(wm ,cross(nm,wm))));
  end
end
bm_hat=cross(wm,cross(rtm,wm))/norm(cross(wm,cross(rtm,wm)));
bm_norm=sqrt(radiusm^2+2*mum*radiusm/vamhe_norm^2);
bm=bm_hat*bm_norm;
%
 He=acosh( (mue+RSOIe*vdehe_norm^2)/(mue+radiuse*vdehe_norm^2));
 deltate=((mue+radiuse*vdehe_norm^2)*sinh(He)-mue*He)/vdehe_norm^3
/24/3600;
 Hm=acosh( (mum+RSOIm*vamhe_norm^2)/(mum+radiusm*vamhe_norm^2));
 deltatm=((mum+radiusm*vamhe_norm^2)*sinh(Hm)-mum*Hm)/vamhe_norm^3
/24/3600;
tpatche=tstart+deltate;
tpatchm=tfinish-deltatm;
[rearth,rmars,vearth,vmars]=EarthMars([tstart tfinish tpatche
tpatchm]);
rpatche=rearth(:,3)+be+we*sqrt(RSOIe^2-be_norm^2);
rpatchm=rmars(:,4)+bm+wm*sqrt(RSOIm^2-bm_norm^2);
if mode=='D';
[v0,v]= DirectIntercept(rpatche,vearth(:,3),rpatchm,tpatchm-
tpatche,mu);deltavmcc=zeros(3,1);
elseif mode=='R';
[v0,v]= RetrogradeIntercept(rpatche,vearth(:,3),rpatchm,tpatchm-
tpatche,mu);deltavmcc=zeros(3,1);
else
[v0,deltavmcc,v]= FimpleIntercept(rpatche,vearth(:,3),rpatchm,tpatchm-
tpatche,mu);
end
vdehe=v0-vearth(:,3);vdehe_norm=norm(vdehe);
vde_norm=sqrt(2*mue/(radiuse)+vdehe_norm^2);
vamhe=v-vmars(:,4);
vamhe_norm=norm(vamhe);
```

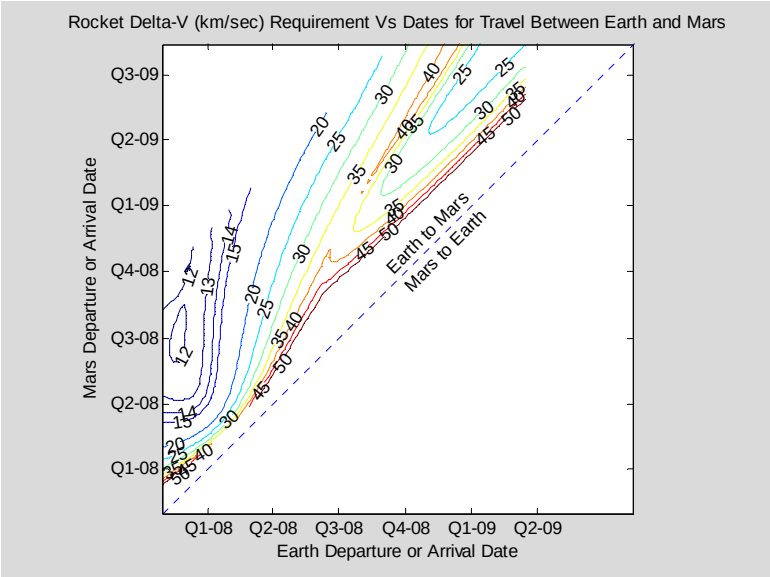
```
vam_norm=sqrt(2*mum/(radiusm)+vamhe_norm^2);
deltav=vde_norm+norm(deltavmcc);
end
DeltaVRocket(istart,jtransfer)=deltav
end
end
contour(DeltaVRocket)
```

```
[XXX,YYY]=meshgrid(tearth,ttransfer);
[CC,hh]=contour(XXX,YYY,DeltaVRocket',[12 13 14 15 20 25 30 35 40 45
50]);
datetick('x');clabel(CC,hh);
xlabel('Earth Departure Date');
ylabel('Number of Days In Flight');
title('Velocity Change (km/sec) for Earth-to-Mars Transfer Vs Departure
Date & Travel Time')
```




```
[XXX,YYY]=meshgrid(tearth,ttransfer);
YYY=YYY+XXX;
tmax=max(max(max(XXX)'),max(max(YYY)'));
tmin=min(min(min(XXX)'),min(min(YYY)'));
[CC,hh]=contour(XXX,YYY,DeltaVRocket',[12 13 14 15 20 25 30 35 40 45
50]);
datetick('x');datetick('y');
axis square; xlim([tmin tmax-180]);ylim([tmin tmax-180]);
tmid=(tmin+tmax-180)/2;
clabel(CC,hh);
```

```
xlabel('Earth Departure or Arrival Date');
ylabel('Mars Departure or Arrival Date');
title('Rocket Delta-V (km/sec) Requirement Vs Dates for Travel Between
Earth and Mars'),
hold on; plot([tmin tmax],[tmin tmax],':');hold off;text(tmid-
15,tmid+15,'Earth to Mars','Rotation',45);text(tmid+10,tmid-10,'Mars to
Earth','Rotation',45),
```



Next, the earth-mars patched conic calculation is put into an M-file:

```
function [deltav_rocket,deltav_reentry]=Earth2Mars(tstart,tfinish);
nm =[0.446158726935355  -0.0554711783756754  0.893233081983105]';
ne =[0.  0.39781867566908  0.917464059943984]';
mus=1.327151544E11;
mue=3.986e5;
mum=4.305e4;
mu=mus;
radiuse=6378;radiusm=3380;
% Calculate heliocentric transfer trajectories between planet centers.
[rearth,rmars,vearth,vmars]=EarthMars([tstart tfinish]);
rstart=rearth(:,1);
rfinish=rmars(:,2);
vstart=vearth(:,1);
deltat=tfinish-tstart;
%  Laplace's recommended "Sphere-Of-Influence" sizes
RSOIe=norm(rearth(:,1))*(mue/mus)^0.4;
```

```
RSOIm=norm(rmars(:,2))*(mum/mus)^0.4;
%
[v0D,vD]= DirectIntercept(rstart,vstart,rfinish,deltat,mu);
vdeheD=v0D-vearth(:,1);
vdeheD_norm=norm(vdeheD);
vdeD_norm=sqrt(2*mue/(radiuse)+vdeheD_norm^2);
vamheD=vD-vmars(:,2);
vamheD_norm=norm(vamheD);
vamD_norm=sqrt(2*mum/(radiusm)+vamheD_norm^2);
%
[v0R,vR]= RetrogradeIntercept(rstart,vstart,rfinish,deltat,mu);
vdeheR=v0R-vearth(:,1);
vdeheR_norm=norm(vdeheR);
vdeR_norm=sqrt(2*mue/(radiuse)+vdeheR_norm^2);
vamheR=vR-vmars(:,2);
vamheR_norm=norm(vamheR);
vamR_norm=sqrt(2*mum/(radiusm)+vamheR_norm^2);
%
[v0F,deltavmcc,vF]= FimpleIntercept(rstart,vstart,rfinish,deltat,mu);
vdeheF=v0F-vearth(:,1);
vdeheF_norm=norm(vdeheF);
vdeF_norm=sqrt(2*mue/(radiuse)+vdeheF_norm^2);
vamheF=vF-vmars(:,2);
vamheF_norm=norm(vamheF);
vamF_norm=sqrt(2*mum/(radiusm)+vamheF_norm^2);
%
deltavD=vdeD_norm;
deltavR=vdeR_norm;
deltavF=vdeF_norm+norm(deltavmcc);
if deltavF<=min(deltavD,deltavR);
 mode='F';
 vdehe=vdeheF;vdehe_norm=vdeheF_norm;vde_norm=vdeF_norm;
 vamhe=vamheF;vamhe_norm=vamheF_norm;vam_norm=vamF_norm;
elseif deltavD<min(deltavF,deltavR);
 mode='D';
 vdehe=vdeheD;vdehe_norm=vdeheD_norm;vde_norm=vdeD_norm;
 vamhe=vamheD;vamhe_norm=vamheD_norm;vam_norm=vamD_norm;
else;
 mode='R';
 vdehe=vdeheR;vdehe_norm=vdeheR_norm;vde_norm=vdeR_norm;
 vamhe=vamheR;vamhe_norm=vamheR_norm;vam_norm=vamR_norm;
end
%
% The "patched conic" loop follows:
for kcount=1:3;  %Three iterations results in adequate accuracy.
```

```
% Determine patch points in planetocentric coordinates.
 etae=acos(mue/(mue+radiuse*vdehe_norm^2));
 we=+vdehe/vdehe_norm;
 se=-rearth(:,1)/norm(rearth(:,1));
 if (norm(cross(ne,we))==0);
  rte=radiuse*(we*cos(etae)+sin(etae)*(se-dot(se,we)*we)/norm(se-dot(se,we)*we));
 elseif abs(dot(ne,we))<sin(etae);
  if dot(se,cross(ne,we))>0;
   rte=radiuse*(+cross(ne,we)/norm(cross(ne,we))*sin(etae)-
cross(ne,cross(ne,we))/norm(cross(ne,cross(ne,we)))*cos(etae));
  else
   rte=radiuse*(-cross(ne,we)/norm(cross(ne,we))*sin(etae)-
cross(ne,cross(ne,we))/norm(cross(ne,cross(ne,we)))*cos(etae));
  end
 else
  if dot(ne,we)>0;
rte=radiuse*(we*cos(etae)+sin(etae)*cross(we,cross(ne,we))/norm(cross(we,cross(ne,we)
)));
  else
rte=radiuse*(we*cos(etae)-sin(etae)*cross(we,cross(ne,we))/norm(cross(we
,cross(ne,we))));
  end
end
be_hat=cross(we,cross(rte,we))/norm(cross(we,cross(rte,we)));
be_norm=sqrt(radiuse^2+2*mue*radiuse/vdehe_norm^2);
be=be_hat*be_norm;
%
 etam=acos(mum/(mum+radiusm*vamhe_norm^2));
 wm=-vamhe/vamhe_norm;
 sm=-rmars(:,2)/norm(rmars(:,2));
 if (norm(cross(nm,wm))==0)
  rtm=radiusm*(wm*cos(etam)+sin(etam)*(sm-dot(sm,wm)*wm)/norm(sm-
dot(sm,wm)*wm));
 elseif abs(dot(nm,wm))<sin(etam);
  if dot(sm,cross(nm,wm))>0;
   rtm=radiusm*(+cross(nm,wm)/norm(cross(nm,wm))*sin(etam)-
cross(nm,cross(nm,wm))/norm(cross(nm,cross(nm,wm)))*cos(etam));
  else
   rtm=radiusm*(-cross(nm,wm)/norm(cross(nm,wm))*sin(etam)-
cross(nm,cross(nm,wm))/norm(cross(nm,cross(nm,wm)))*cos(etam));
  end
 else
  if dot(nm,wm)>0;
rtm=radiusm*(wm*cos(etam)+sin(etam)*cross(wm,cross(nm,wm))/norm(cross(wm,cross
(nm,wm))));
```

```
  else
rtm=radiusm*(wm*cos(etam)-sin(etam)*cross(wm,cross(nm,wm))/norm(cross(wm
,cross(nm,wm))));
  end
end
bm_hat=cross(wm,cross(rtm,wm))/norm(cross(wm,cross(rtm,wm)));
bm_norm=sqrt(radiusm^2+2*mum*radiusm/vamhe_norm^2);
bm=bm_hat*bm_norm;
%
 He=acosh( (mue+RSOIe*vdehe_norm^2)/(mue+radiuse*vdehe_norm^2));
 deltate=((mue+radiuse*vdehe_norm^2)*sinh(He)-mue*He)/vdehe_norm^3 /24/3600;
 Hm=acosh( (mum+RSOIm*vamhe_norm^2)/(mum+radiusm*vamhe_norm^2));
 deltatm=((mum+radiusm*vamhe_norm^2)*sinh(Hm)-mum*Hm)/vamhe_norm^3
/24/3600;
tpatche=tstart+deltate;
tpatchm=tfinish-deltatm;
[rearth,rmars,vearth,vmars]=EarthMars([tstart tfinish tpatche tpatchm]);
rpatche=rearth(:,3)+be+we*sqrt(RSOIe^2-be_norm^2);
rpatchm=rmars(:,4)+bm+wm*sqrt(RSOIm^2-bm_norm^2);
if mode=='D';
[v0,v]= DirectIntercept(rpatche,vearth(:,3),rpatchm,tpatchm-
tpatche,mu);deltavmcc=zeros(3,1);
elseif mode=='R';
[v0,v]= RetrogradeIntercept(rpatche,vearth(:,3),rpatchm,tpatchm-
tpatche,mu);deltavmcc=zeros(3,1);
else
[v0,deltavmcc,v]= FimpleIntercept(rpatche,vearth(:,3),rpatchm,tpatchm-tpatche,mu);
end
vdehe=v0-vearth(:,3);vdehe_norm=norm(vdehe);
vde_norm=sqrt(2*mue/(radiuse)+vdehe_norm^2);
vamhe=v-vmars(:,4);
vamhe_norm=norm(vamhe);
vam_norm=sqrt(2*mum/(radiusm)+vamhe_norm^2);
deltav_rocket=vde_norm+norm(deltavmcc);
deltav_reentry=vam_norm;
end
```

Then the m-file is exercised:

```
tstart=datenum('15-Aug-2006');tfinish=datenum('1-August-2008');
[deltav_rocket,deltav_reentry]=Earth2Mars(tstart,tfinish)

deltav_rocket =
   12.6958
deltav_reentry =
```

```
    9.9639
```

Similarly, the inverse transfer (i.e., from mars to earth) is as follows:

```
function [deltav_rocket,deltav_reentry]=Mars2Earth(tstart,tfinish);
nm =[0.446158726935355  -0.0554711783756754  0.893233081983105]';
ne =[0.  0.39781867566908  0.917464059943984]';
mus=1.327151544E11;
mue=3.986e5;
mum=4.305e4;
mu=mus;
radiuse=6378;radiusm=3380;
% Calculate heliocentric transfer trajectories between planet centers.
[rearth,rmars,vearth,vmars]=EarthMars([tstart tfinish]);
rstart=rmars(:,1);
rfinish=rearth(:,2);
vstart=vmars(:,1);
deltat=tfinish-tstart;
%  Laplace's recommended "Sphere-Of-Influence" sizes
RSOIe=norm(rearth(:,2))*(mue/mus)^0.4;
RSOIm=norm(rmars(:,1))*(mum/mus)^0.4;
%
[v0D,vD]= DirectIntercept(rstart,vstart,rfinish,deltat,mu);
vdmheD=v0D-vmars(:,1);
vdmheD_norm=norm(vdmheD);
vdmD_norm=sqrt(2*mum/(radiusm)+vdmheD_norm^2);
vaeheD=vD-vearth(:,2);
vaeheD_norm=norm(vaeheD);
vaeD_norm=sqrt(2*mue/(radiuse)+vaeheD_norm^2);
%
[v0R,vR]= RetrogradeIntercept(rstart,vstart,rfinish,deltat,mu);
vdmheR=v0R-vmars(:,1);
vdmheR_norm=norm(vdmheR);
vdmR_norm=sqrt(2*mum/(radiusm)+vdmheR_norm^2);
vaeheR=vR-vearth(:,2);
vaeheR_norm=norm(vaeheR);
vaeR_norm=sqrt(2*mue/(radiuse)+vaeheR_norm^2);
%
[v0F,deltavmcc,vF]= FimpleIntercept(rstart,vstart,rfinish,deltat,mu);
vdmheF=v0F-vmars(:,1);
vdmheF_norm=norm(vdmheF);
vdmF_norm=sqrt(2*mum/(radiusm)+vdmheF_norm^2);
vaeheF=vF-vearth(:,2);
vaeheF_norm=norm(vaeheF);
```

```
vaeF_norm=sqrt(2*mue/(radiuse)+vaeheF_norm^2);
%
deltavD=vdmD_norm;
deltavR=vdmR_norm;
deltavF=vdmF_norm+norm(deltavmcc);
if deltavF<=min(deltavD,deltavR);
 mode='F';
 vdmhe=vdmheF;vdmhe_norm=vdmheF_norm;vdm_norm=vdmF_norm;
 vaehe=vaeheF;vaehe_norm=vaeheF_norm;vae_norm=vaeF_norm;
elseif deltavD<min(deltavF,deltavR);
 mode='D';
 vdmhe=vdmheD;vdmhe_norm=vdmheD_norm;vdm_norm=vdmD_norm;
 vaehe=vaeheD;vaehe_norm=vaeheD_norm;vae_norm=vaeD_norm;
else;
 mode='R';
 vdmhe=vdmheR;vdmhe_norm=vdmheR_norm;vdm_norm=vdmR_norm;
 vaehe=vaeheR;vaehe_norm=vaeheR_norm;vae_norm=vaeR_norm;
end
%
% The "patched conic" loop follows:
for kcount=1:3;  %Three iterations results in adequate accuracy.
% Determine patch points in planetocentric coordinates.
 etae=acos(mue/(mue+radiuse*vaehe_norm^2));
 we=-vaehe/vaehe_norm;
 se=-rearth(:,2)/norm(rearth(:,2));
 if (norm(cross(ne,we))==0);
  rte=radiuse*(we*cos(etae)+sin(etae)*(se-dot(se,we)*we)/norm(se-dot(se,we)*we));
 elseif abs(dot(ne,we))<sin(etae);
  if dot(se,cross(ne,we))>0;
   rte=radiuse*(+cross(ne,we)/norm(cross(ne,we))*sin(etae)-
cross(ne,cross(ne,we))/norm(cross(ne,cross(ne,we)))*cos(etae));
  else
   rte=radiuse*(-cross(ne,we)/norm(cross(ne,we))*sin(etae)-
cross(ne,cross(ne,we))/norm(cross(ne,cross(ne,we)))*cos(etae));
  end
 else
  if dot(ne,we)>0;
rte=radiuse*(we*cos(etae)+sin(etae)*cross(we,cross(ne,we))/norm(cross(we,cross(ne,we)
)));
  else
rte=radiuse*(we*cos(etae)-sin(etae)*cross(we,cross(ne,we))/norm(cross(we
,cross(ne,we))));
  end
end
be_hat=cross(we,cross(rte,we))/norm(cross(we,cross(rte,we)));
```

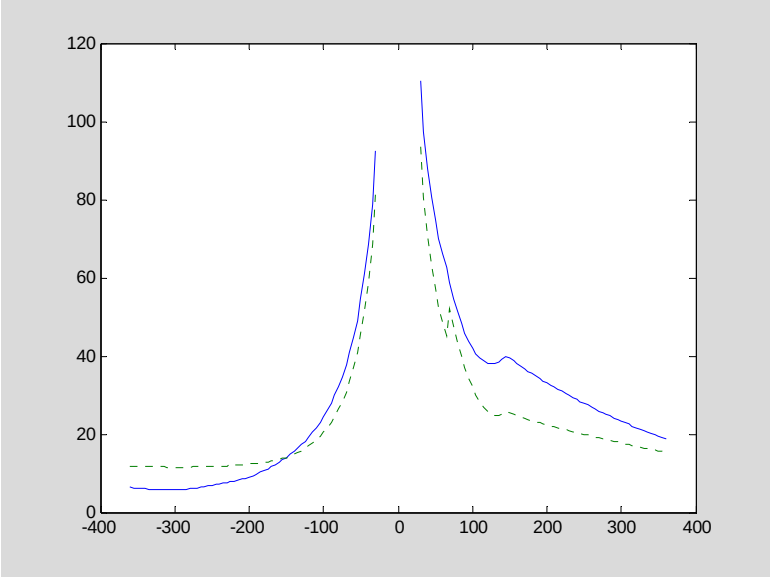
```
be_norm=sqrt(radiuse^2+2*mue*radiuse/vaehe_norm^2);
be=be_hat*be_norm;
%
 etam=acos(mum/(mum+radiusm*vdmhe_norm^2));
 wm=+vdmhe/vdmhe_norm;
 sm=-rmars(:,1)/norm(rmars(:,1));
 if (norm(cross(nm,wm))==0);
  rtm=radiusm*(wm*cos(etam)+sin(etam)*(sm-dot(sm,wm)*wm)/norm(sm-
dot(sm,wm)*wm));
 elseif abs(dot(nm,wm))<sin(etam);
  if dot(sm,cross(nm,wm))>0;
   rtm=radiusm*(+cross(nm,wm)/norm(cross(nm,wm))*sin(etam)-
cross(nm,cross(nm,wm))/norm(cross(nm,cross(nm,wm)))*cos(etam));
  else
   rtm=radiusm*(-cross(nm,wm)/norm(cross(nm,wm))*sin(etam)-
cross(nm,cross(nm,wm))/norm(cross(nm,cross(nm,wm)))*cos(etam));
  end
 else
  if dot(nm,wm)>0;
rtm=radiusm*(wm*cos(etam)+sin(etam)*cross(wm,cross(nm,wm))/norm(cross(wm,cross
(nm,wm))));
  else
rtm=radiusm*(wm*cos(etam)-sin(etam)*cross(wm,cross(nm,wm))/norm(cross(wm
,cross(nm,wm))));
  end
end
bm_hat=cross(wm,cross(rtm,wm))/norm(cross(wm,cross(rtm,wm)));
bm_norm=sqrt(radiusm^2+2*mum*radiusm/vdmhe_norm^2);
bm=bm_hat*bm_norm;
%
 He=acosh( (mue+RSOIe*vaehe_norm^2)/(mue+radiuse*vaehe_norm^2));
 deltate=((mue+radiuse*vaehe_norm^2)*sinh(He)-mue*He)/vaehe_norm^3 /24/3600;
 Hm=acosh( (mum+RSOIm*vdmhe_norm^2)/(mum+radiusm*vdmhe_norm^2));
 deltatm=((mum+radiusm*vdmhe_norm^2)*sinh(Hm)-mum*Hm)/vdmhe_norm^3
/24/3600;
tpatchm=tstart+deltatm;
tpatche=tfinish-deltate;
[rearth,rmars,vearth,vmars]=EarthMars([tstart tfinish tpatchm tpatche]);
rpatche=rearth(:,4)+be+we*sqrt(RSOIe^2-be_norm^2);
rpatchm=rmars(:,3)+bm+wm*sqrt(RSOIm^2-bm_norm^2);
if mode=='D';
[v0,v]= DirectIntercept(rpatchm,vmars(:,3),rpatche,tpatche-
tpatchm,mu);deltavmcc=zeros(3,1);
elseif mode=='R';
```

```
[v0,v]= RetrogradeIntercept(rpatchm,vmars(:,3),rpatche,tpatche-
tpatchm,mu);deltavmcc=zeros(3,1);
else
[v0,deltavmcc,v]= FimpleIntercept(rpatchm,vmars(:,3),rpatche,tpatche-tpatchm,mu);
end
vaehe=v-vearth(:,4);vaehe_norm=norm(vaehe);
vae_norm=sqrt(2*mue/(radiuse)+vaehe_norm^2);
vdmhe=v0-vmars(:,3);vdmhe_norm=norm(vdmhe);
vdm_norm=sqrt(2*mum/(radiusm)+vdmhe_norm^2);
deltav_rocket=vdm_norm+norm(deltavmcc);
deltav_reentry=vae_norm;
end
```

Then the M-file is exercised:

```
tstart=datenum('1-November-2007');tfinish=datenum('15-July-2008');
[deltav_rocket,deltav_reentry]=Mars2Earth(tstart,tfinish)

deltav_rocket =
    6.9027
deltav_reentry =
   11.9224
```

```
ttransfer=([-360:5:-30 0 30:5:360])';
mdata=size(ttransfer,1);
deltav_rocket=zeros(mdata,1);deltav_reentry=zeros(mdata,1);
tearth=datenum('1-July-2008');
for i=1:mdata;
 if ttransfer(i)<0;
[deltav_rocket(i),deltav_reentry(i)]=Mars2Earth(ttransfer(i)+tearth,tea
rth);
 elseif ttransfer(i)>0;
[deltav_rocket(i),deltav_reentry(i)]=Earth2Mars(tearth,tearth+ttransfer
(i));
else
  deltav_rocket(i)=NaN; deltav_reentry(i)=NaN;
 end
end
plot(ttransfer,deltav_rocket,ttransfer,deltav_reentry,':')
```




```
DeltaV_Rocket=[];DeltaV_Reentry=[];
tbegin=datenum('1-Jan-2000');tend=datenum('1-March-2000');
Tearth=[];
ttransfer=([-360:5:-30 0 30:5:360])';
mdata=size(ttransfer,1);
%
while isempty(Tearth) | Tearth(end)<tend;
deltav_rocket=zeros(mdata,1);deltav_reentry=zeros(mdata,1);
if isempty(Tearth)
 Tearth=tbegin;
else
 Tearth=[Tearth (Tearth(end)+5)]
end
tearth=Tearth(end);
for i=1:mdata;
 if ttransfer(i)<0;
[deltav_rocket(i),deltav_reentry(i)]=Mars2Earth(ttransfer(i)+tearth,tea
rth);
 elseif ttransfer(i)>0;
[deltav_rocket(i),deltav_reentry(i)]=Earth2Mars(tearth,tearth+ttransfer
(i));
else
  deltav_rocket(i)=NaN; deltav_reentry(i)=NaN;
end
end
DeltaV_Rocket=[DeltaV_Rocket deltav_rocket];
DeltaV_Reentry=[DeltaV_Reentry deltav_reentry];
end
'done'

Tearth =
      730486      730491
Tearth =
```

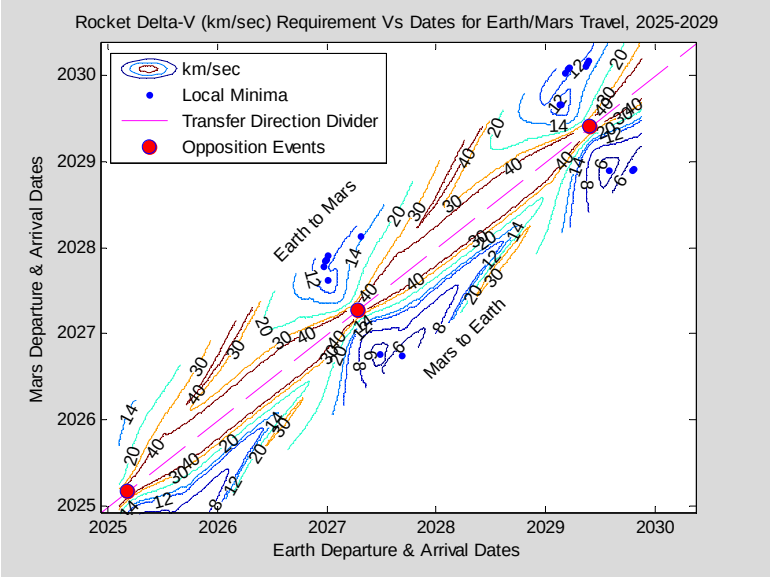
```
      730486      730491      730496
Tearth =
      730486      730491      730496      730501
Tearth =
      730486      730491      730496      730501      730506
Tearth =
      730486      730491      730496      730501      730506
730511
Tearth =
  Columns 1 through 6
      730486      730491      730496      730501      730506
730511
  Column 7
      730516
Tearth =
  Columns 1 through 6
      730486      730491      730496      730501      730506
730511
  Columns 7 through 8
      730516      730521
Tearth =
  Columns 1 through 6
      730486      730491      730496      730501      730506
730511
  Columns 7 through 9
      730516      730521      730526
Tearth =
  Columns 1 through 6
      730486      730491      730496      730501      730506
730511
  Columns 7 through 10
      730516      730521      730526      730531
Tearth =
  Columns 1 through 6
      730486      730491      730496      730501      730506
730511
  Columns 7 through 11
      730516      730521      730526      730531      730536
Tearth =
  Columns 1 through 6
      730486      730491      730496      730501      730506
730511
  Columns 7 through 12
      730516      730521      730526      730531      730536
730541
Tearth =
  Columns 1 through 6
      730486      730491      730496      730501      730506
730511
  Columns 7 through 12
      730516      730521      730526      730531      730536
730541
  Column 13
      730546
ans =
```

```
done



load OrbitalTransferData
whos

  Name                    Size                    Bytes  Class

  CC                      2x6569                 105104  double array
  D_e_m                   1x1000                   8000  double array
  DeltaV                  1x50                      400  double array
  DeltaVMultiple          1x50                      400  double array
  DeltaVMultiple10        1x50                      400  double array
  DeltaVMultiple6         1x50                      400  double array
  DeltaV_CH4              1x50                      400  double array
  DeltaV_H2               1x50                      400  double array
  DeltaV_Kerosene         1x50                      400  double array
  DeltaV_Reentry        135x7306                7890480  double array
  DeltaV_Rocket         135x7306                7890480  double array
  III                     1x12                       96  double array
  PayloadFraction         1x50                      400  double array
  Tearth                  1x7306                  58448  double array
  Topposition             1x12                       96  double array
  XXX                   135x351                  379080  double array
  YYY                   135x351                  379080  double array
  hh                      1x1                         8  double array
  i                       1x1                         8  double array
  rearth                  3x1000                  24000  double array
  rmars                   3x1000                  24000  double array
  t                       1x1000                   8000  double array
  tmax                    1x1                         8  double array
  tmid                    1x1                         8  double array
  tmin                    1x1                         8  double array
  ttransfer             135x1                      1080  double array
  vearth                  3x1000                  24000  double array
  vmars                   3x1000                  24000  double array
  xqxqxq1234              1x1                         1  logical array
  xqxqxq1235              1x3                         6  char array

Grand total is 2102402 elements using 16819191 bytes

DeltaV_Rocket=DeltaV_Rocket(:,(0.251*7306):(0.299*7306));
DeltaV_Reentry=DeltaV_Reentry(:,(0.251*7306):(0.299*7306));
Tearth=Tearth((0.251*7306):(0.299*7306)); whos

Warning: Integer operands are required for colon operator when used as
index.
Warning: Integer operands are required for colon operator when used as
index.
Warning: Integer operands are required for colon operator when used as
index.
  Name                    Size                    Bytes  Class
```

```
  CC                    2x6569                105104  double array
  D_e_m                 1x1000                 8000  double array
  DeltaV                1x50                    400  double array
  DeltaVMultiple        1x50                    400  double array
  DeltaVMultiple10      1x50                    400  double array
  DeltaVMultiple6       1x50                    400  double array
  DeltaV_CH4            1x50                    400  double array
  DeltaV_H2             1x50                    400  double array
  DeltaV_Kerosene       1x50                    400  double array
  DeltaV_Reentry      135x351                379080  double array
  DeltaV_Rocket       135x351                379080  double array
  III                   1x12                    96  double array
  PayloadFraction       1x50                    400  double array
  Tearth                1x351                  2808  double array
  Topposition           1x12                    96  double array
  XXX                 135x351                379080  double array
  YYY                 135x351                379080  double array
  hh                    1x1                        8  double array
  i                     1x1                        8  double array
  rearth                3x1000                 24000  double array
  rmars                 3x1000                 24000  double array
  t                     1x1000                  8000  double array
  tmax                  1x1                        8  double array
  tmid                  1x1                        8  double array
  tmin                  1x1                        8  double array
  ttransfer           135x1                    1080  double array
  vearth                3x1000                 24000  double array
  vmars                 3x1000                 24000  double array
  xqxqxq1234            1x1                        1  logical array
  xqxqxq1235            1x3                        6  char array

Grand total is 217597 elements using 1740751 bytes
```

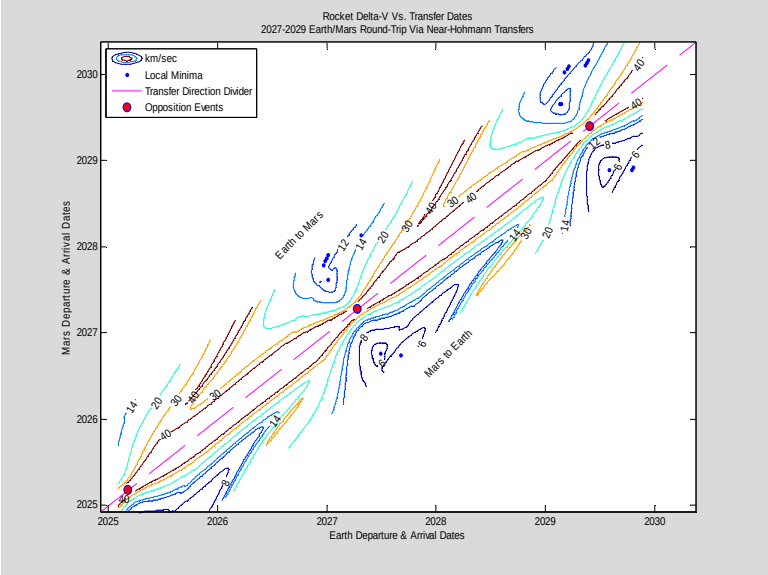
```
[m,n]=size(DeltaV_Rocket),
[II,JJ]=find([zeros(m,1) (diff((diff(DeltaV_Rocket,1,2)<0),1,2)<0)
zeros(m,1)] & [zeros(1,n); (diff((diff(DeltaV_Rocket,1,1)<0),1,1)<0) ;
zeros(1,n)]);Indices=[II JJ];
disp(['Earth         Mars      DeltaV Rocket']),
disp([datestr(Tearth(Indices(:,2))) blanks(23)' blanks(23)'
datestr(Tearth(Indices(:,2))+ttransfer(Indices(:,1))') blanks(23)'
blanks(23)' num2str(diag(DeltaV_Rocket(Indices(:,1),Indices(:,2)))) ])

m =
   135
n =
   351
Earth         Mars      DeltaV Rocket
08-Jun-2025  06-Sep-2024  5.72608
28-Jun-2025  27-Aug-2024  5.72631
03-Jul-2025  27-Aug-2024  5.72082
08-Jul-2025  27-Aug-2024  5.71886
20-Dec-2026  16-Oct-2027  11.5749
```

```
25-Dec-2026  05-Nov-2027  11.5699
30-Dec-2026  15-Nov-2027  11.5712

04-Jan-2027  17-Aug-2027  11.5753

04-Jan-2027  30-Nov-2027  11.5749
24-Apr-2027  23-Feb-2028   13.642
28-Jun-2027  06-Oct-2026  5.65563
06-Sep-2027  01-Oct-2026  5.63109
17-Feb-2029  31-Aug-2029  11.5793
22-Feb-2029  31-Aug-2029  11.5766
04-Mar-2029  13-Jan-2030  11.5501
14-Mar-2029  28-Jan-2030  11.5468
19-Mar-2029  07-Feb-2030  11.5464
13-May-2029  12-Feb-2030  12.5019
18-May-2029  22-Feb-2030  12.4911
23-May-2029  04-Mar-2030  12.4689

01-Aug-2029  24-Nov-2028  5.59386

15-Oct-2029  24-Nov-2028  5.56339
20-Oct-2029  04-Dec-2028  5.56303

'LineWidth',2,


clf;
[XXX,YYY]=meshgrid(Tearth,ttransfer);
YYY=YYY+XXX;
tmax=max(max(max(XXX)'),max(max(YYY)'));
tmin=min(min(min(XXX)'),min(min(YYY)'));
[CC,hh]=contour(XXX,YYY,DeltaV_Rocket,[6 8 12 14 20 30 40]);
datetick('x'),datetick('y'),
xlim([tmin+300 tmax-180]);ylim([tmin+300 tmax-180]);
tmid=(tmin+tmax)/2;
clabel(CC,hh);
xlabel('Earth Departure & Arrival Dates'), ylabel('Mars Departure &
Arrival Dates'),title('Rocket Delta-V (km/sec) Requirement Vs Dates for
Earth/Mars Travel, 2025-2029');
hold on;
plot(Tearth(Indices(:,2)),Tearth(Indices(:,2))+ttransfer(Indices(:,1))'
,'.',[tmin tmax],[tmin tmax], '--m',
Topposition,Topposition,'o','MarkerEdgeColor','b','MarkerFaceColor','r'
,'MarkerSize',8),legend('km/sec','Local Minima','Transfer Direction
Divider','Opposition Events','Location','NorthWest');hold
off;text(tmid-100-250,tmid-100+250,'Earth to
Mars','Rotation',45);text(tmid-100+250,tmid-100-250,'Mars to
Earth','Rotation',45)
```



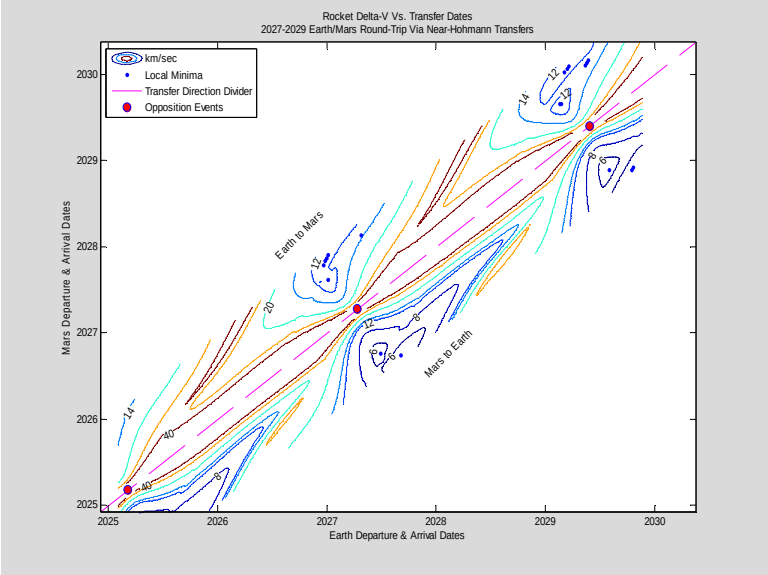

```
clf;
[XXX,YYY]=meshgrid(Tearth,ttransfer);
YYY=YYY+XXX;
tmax=max(max(max(XXX)'),max(max(YYY)'));
tmin=min(min(min(XXX)'),min(min(YYY)'));
[CC,hh]=contour(XXX,YYY,DeltaV_Rocket,[6 8 12 14 20 30 40]);
datetick('x'),datetick('y'),
xlim([tmin+300 tmax-180]);ylim([tmin+300 tmax-180]);
tmid=(tmin+tmax)/2;
clabel(CC,hh,'manual');
xlabel('Earth Departure & Arrival Dates'), ylabel('Mars Departure &
Arrival Dates'),title({'Rocket Delta-V Vs. Transfer Dates'; '2027-2029
Earth/Mars Round-Trip Via Near-Hohmann Transfers'});
hold on;
plot(Tearth(Indices(:,2)),Tearth(Indices(:,2))+ttransfer(Indices(:,1))'
,'.',[tmin tmax],[tmin tmax], '--m',
Topposition,Topposition,'o','MarkerEdgeColor','b','MarkerFaceColor','r'
,'MarkerSize',8),legend('km/sec','Local Minima','Transfer Direction
Divider','Opposition Events','Location','NorthWest');hold
off;text(tmid-100-250,tmid-100+250,'Earth to
Mars','Rotation',45);text(tmid-100+250,tmid-100-250,'Mars to
Earth','Rotation',45)
```

```
    Please wait a moment...

   Carefully select contours for labeling.
   When done, press RETURN while the Graph window is the active window.
```

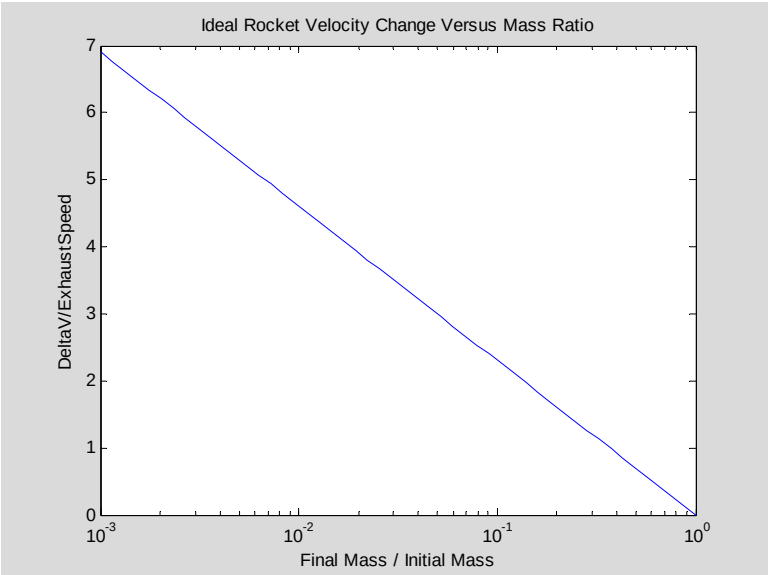



```
clf;
[XXX,YYY]=meshgrid(Tearth,ttransfer);
YYY=YYY+XXX;
tmax=max(max(max(XXX)'),max(max(YYY)'));
tmin=min(min(min(XXX)'),min(min(YYY)'));
[CC,hh]=contour(XXX,YYY,DeltaV_Rocket,[6 8 12 14 20 30 40]);
datetick('x'),datetick('y'),
xlim([tmin+300 tmax-180]);ylim([tmin+300 tmax-180]);
tmid=(tmin+tmax)/2;
clabel(CC,hh,'manual');
xlabel('Earth Departure & Arrival Dates'), ylabel('Mars Departure &
Arrival Dates'),title({'Rocket Delta-V Vs. Transfer Dates'; '2027-2029
Earth/Mars Round-Trip Via Near-Hohmann Transfers'});
hold on;
plot(Tearth(Indices(:,2)),Tearth(Indices(:,2))+ttransfer(Indices(:,1))'
,'.',[tmin tmax],[tmin tmax], '--m',
Topposition,Topposition,'o','MarkerEdgeColor','b','MarkerFaceColor','r'
,'MarkerSize',8),legend('km/sec','Local Minima','Transfer Direction
Divider','Opposition Events','Location','NorthWest');hold
off;text(tmid-100-250,tmid-100+250,'Earth to
Mars','Rotation',45);text(tmid-100+250,tmid-100-250,'Mars to
Earth','Rotation',45)
```

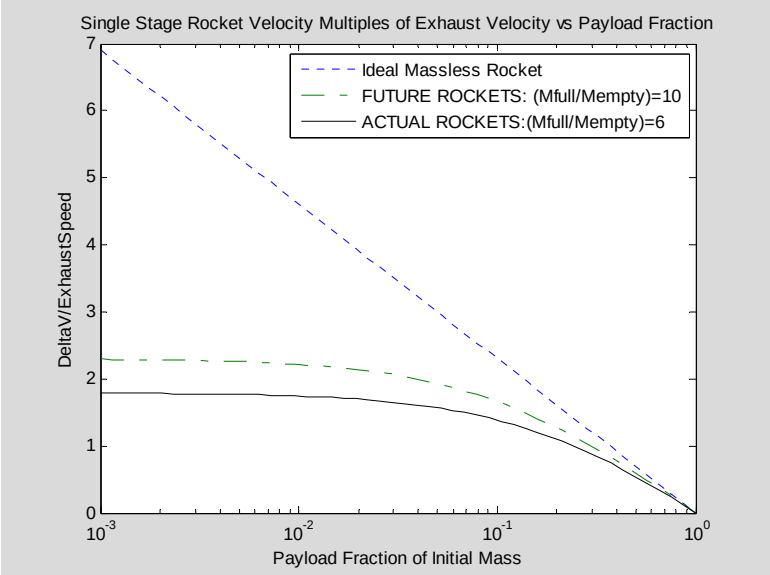
```
    Please wait a moment...

   Carefully select contours for labeling.
   When done, press RETURN while the Graph window is the active window.
```



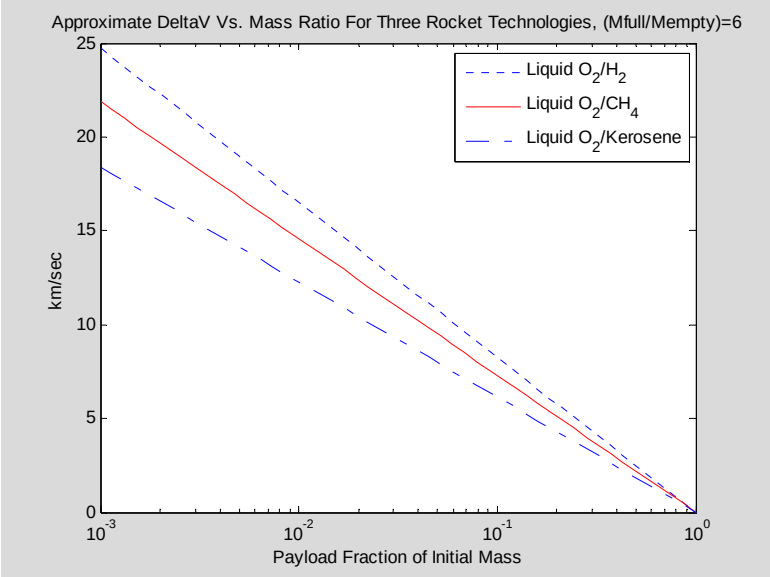

```
PayloadFraction=logspace(-3,0);DeltaVMultiple=-
log(PayloadFraction);semilogx(PayloadFraction,DeltaVMultiple),
xlabel('Final Mass / Initial
Mass'),ylabel('DeltaV/ExhaustSpeed'),title('Ideal Rocket Velocity
Change Versus Mass Ratio')
```



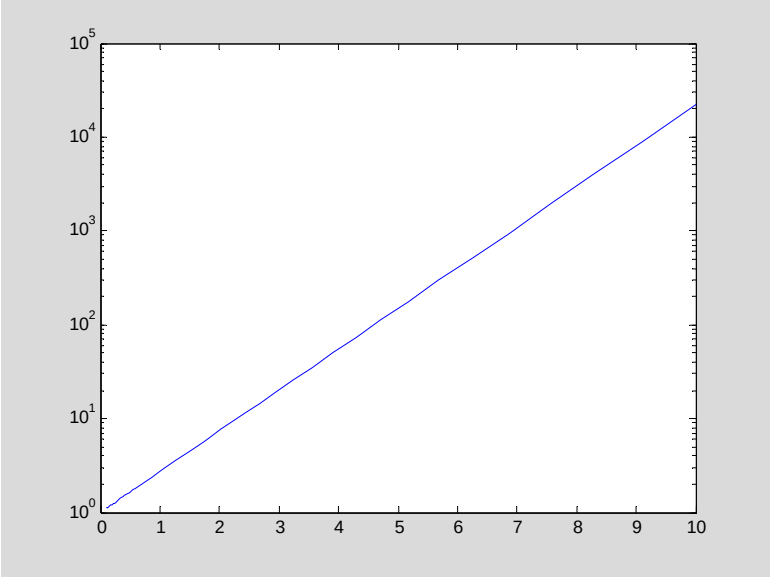
```
DeltaVMultiple6=-log(PayloadFraction+(1-PayloadFraction)/6);
DeltaVMultiple10=-log( PayloadFraction+(1-PayloadFraction)/10);
semilogx(PayloadFraction,DeltaVMultiple,':',PayloadFraction,DeltaVMulti
ple10,'-.',PayloadFraction,DeltaVMultiple6,'k'),xlabel('Payload
Fraction of Initial Mass'),ylabel('DeltaV/ExhaustSpeed'),title('Single
Stage Rocket Velocity Multiples of Exhaust Velocity vs Payload
Fraction'),
legend('Ideal Massless Rocket','FUTURE ROCKETS: (Mfull/Mempty)=10',
'ACTUAL ROCKETS:(Mfull/Mempty)=6','Location','NorthEast')
```




```
DeltaV_H2=-(5/6)*4.30*log(PayloadFraction);
DeltaV_CH4=-(5/6)*3.80*log(PayloadFraction);
DeltaV_Kerosene=-(5/6)*3.20*log(PayloadFraction);
semilogx(PayloadFraction,DeltaV_H2,':',PayloadFraction,DeltaV_CH4,'r',
PayloadFraction,DeltaV_Kerosene,'-.b'),
title('Approximate DeltaV Vs. Mass Ratio For Three Rocket Technologies,
(Mfull/Mempty)=6'),
legend('Liquid O_2/H_2','Liquid O_2/CH_4','Liquid
O_2/Kerosene','Location','NorthEast'),
ylabel('km/sec'),
xlabel('Payload Fraction of Initial Mass')
```




```
SpeedRatio=logspace(-1,1);
MassRatio=exp(SpeedRatio);
semilogy(SpeedRatio,MassRatio)
```



As presented in [Bond and Allman 1996, 76], the following universal variable equations apply. Note that they use the auxiliary variable, *x(t)*.

$$r_0 = \left\| \vec{r}_0 \right\|$$
$$v_0 = \left\| \vec{v}_0 \right\|$$
$$\alpha_0 = \frac{2}{r_0} - \frac{v_0^2}{\mu}$$
$$f(t) = 1 - \frac{x(t)^2}{r_0} C\left(\alpha_0 x(t)^2\right)$$
$$g(t) = t - t_0 - \frac{x(t)^2}{\sqrt{\mu}} S\left(\alpha_0 x(t)^2\right)$$
$$\vec{r}(t) = f(t)\vec{r}_0 + g(t)\vec{v}_0$$

Rocket Energy Efficiency

If we assume that a a single stage rocket with constant exhaust speed starts from zero speed and we neglect both gravity and aerodynamic forces, the total kinetic energy imparted to its non-propellant payload is:

$$E_{\text{payload}} = \frac{1}{2} M_{\text{final}} v^2$$

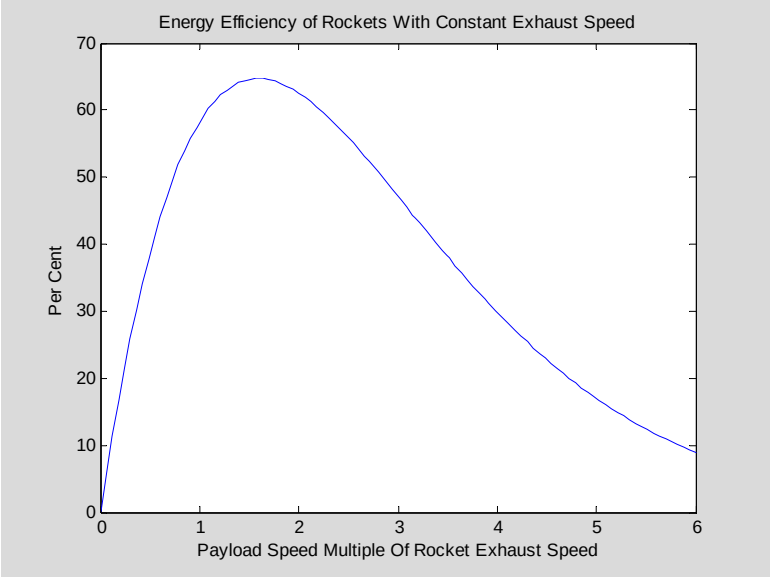
while the kinetic energy invested in propellant is

$$E_{\text{propellant}} = \frac{1}{2}\left(M_{\text{initial}} - M_{\text{final}}\right) u^2$$

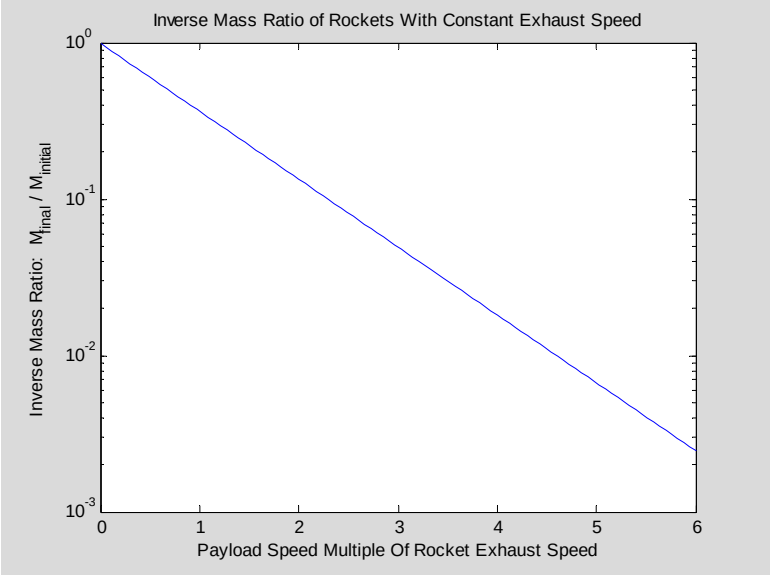
These allow the *rocket energy efficiency* to be defined as

$$\varepsilon_{rocket} = \frac{E_{\text{payload}}}{E_{\text{propellant}}} = \frac{\left(\frac{v}{u}\right)^2}{\exp\left(\frac{v}{u}\right) - 1}$$

```
x=linspace(.0001,6);y=x.^2./(exp(x)-1);plot(x,100*y),title('Energy
Efficiency of Rockets With Constant Exhaust Speed'),xlabel('Payload
Speed Multiple Of Rocket Exhaust Speed'),ylabel('Per Cent')
```

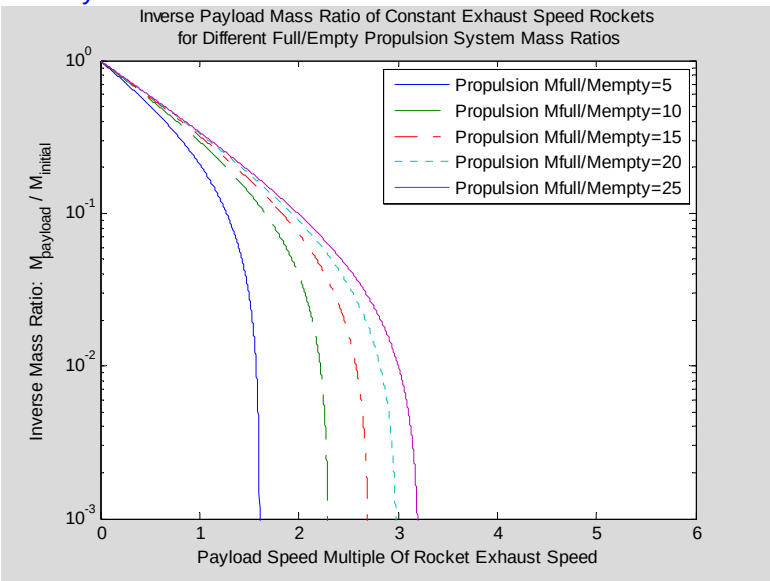



```
InvMassratio=exp(-x);semilogy(x,InvMassratio),title('Inverse Mass Ratio
of Rockets With Constant Exhaust Speed'),ylabel('Inverse Mass Ratio:
M_f_i_n_a_l / M_i_n_i_t_i_a_l' ),xlabel('Payload Speed Multiple Of
Rocket Exhaust Speed')
```




```
x=linspace(0.0001,6,5000);
InvPayloadRatio=zeros(5000,5);FER=[5 10 15 20 25];
InvPayload=(exp(-x')*FER-1)./(ones(5000,1)*FER-1);
semilogy(x,InvPayload(:,1),'-',x,InvPayload(:,2),'--',
x,InvPayload(:,3),'-.', x,InvPayload(:,4),':', x,InvPayload(:,5),'-
'),ylim([1e-3 1]),xlim([0 6]);
title({'Inverse Payload Mass Ratio of Constant Exhaust Speed Rockets','
for Different Full/Empty Propulsion System Mass Ratios'}),
xlabel('Payload Speed Multiple Of Rocket Exhaust Speed'),
```

```
ylabel('Inverse Mass Ratio:  M_p_a_y_l_o_a_d / M_i_n_i_t_i_a_l' )
legend('Propulsion Mfull/Mempty=5','Propulsion
Mfull/Mempty=10','Propulsion Mfull/Mempty=15',  'Propulsion
Mfull/Mempty=20','Propulsion Mfull/Mempty=25','Location','NorthEast')

Warning: Negative data ignored
> In ylim at 44
```



It is sometimes useful to make approximate calculations in which orbits of the Earth and Mars are approximated by coplanar circles. For instance, the Hohmann optimal transfer ius only defined for such an approximate system. To apply this we must have a way to generate the motions of Earth and Mars in this approximation. That is done by the following code:

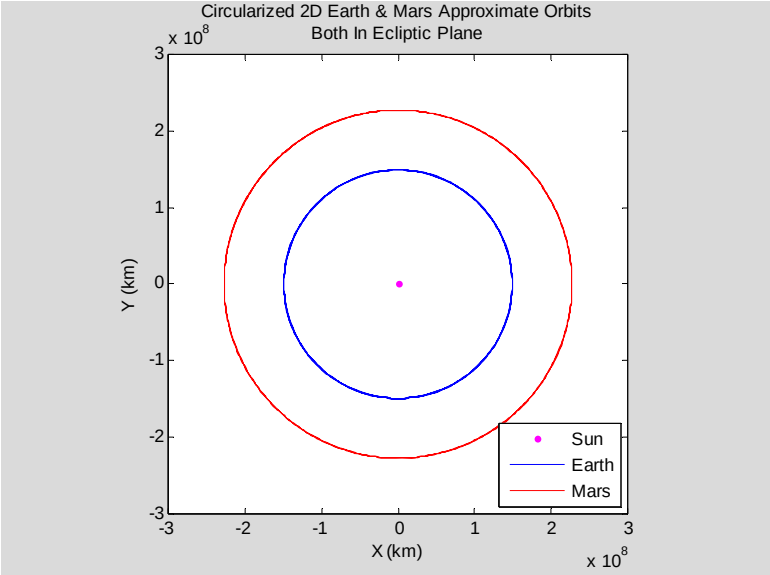
```
function  [rearth,rmars,vearth,vmars]=Circ2D_EarthMars(t)
% Calculates Circular2D Approx.Earth & Mars Position & Velocity Vectors for set of
times.
% Eccentries and Inclinations of orbits are all approximated falsely as zero.
% Coordinate System is standard Heliocentric Cartesian, units km & km/sec
% t is (1 by ) row vector of times using the standard MATLAB time system.
% Outputs are (3 by n) matrices whose columns are vectors at corresponding times.
% Algorithm uses orbital data valid for epoch date at start of calendar year 1990.
%  Coded by Robert Woolley, May 2006
%
daysperyear=365.242191;
kmperAU=149597870.;
```

```
Tp_yrs=[1.00004  1.880932];
incl_deg=[0.  0.];
a_AU=[1.00000  1.523688];
eccentricity=[0.0  0.0];
longt_asc_node_deg=[0  49.480308];
longt_peri_deg=[102.768413  335.874939];
longt_epoch_deg=[99.403308  240.739474];
t_epoch=datenum('00-Jan-1990');
lan=longt_asc_node_deg*pi/180.;
%
trueanomaly_epoch=[longt_epoch_deg-longt_peri_deg]*pi/180.;
E_epoch=acos((eccentricity+cos(trueanomaly_epoch))./(1+eccentricity.*cos(trueanomaly
_epoch)));
M0=eccentricity.*sin(E_epoch)-E_epoch;
p=kmperAU*a_AU.*(1-eccentricity.^2);
arg_peri=(longt_peri_deg-longt_asc_node_deg)*pi/180.;
carg_peri=cos(arg_peri);sarg_peri=sin(arg_peri);
lan=longt_asc_node_deg*pi/180.;
clan=cos(lan);slan=sin(lan);
incl=incl_deg*pi/180.;
cincl=cos(incl);sincl=sin(incl);
% Iterative solution of Kepler's equation:
ecc=ones(size(t'))*eccentricity;
M=ones(size(t'))*M0+2*pi*t'/daysperyear*(1./Tp_yrs);
E=M;
for i=1:10;
 delta=E-ecc.*sin(E)-M;
  if abs(max(max(delta)'))<10*eps
   break
  end
 E=E-delta./(1-ecc.*cos(E));
end
phi=atan2(sqrt(1-ecc.^2).*sin(E)./(1-ecc.*cos(E)),(cos(E)-ecc)./(1-ecc.*cos(E)));
r=ones(size(t'))*p./(1+ecc.*cos(phi));
%
P_hat=zeros(3,2);Q_hat=P_hat;
P_hat(1,:)=clan.*carg_peri-slan.*cincl.*sarg_peri;
P_hat(2,:)=slan.*carg_peri+clan.*cincl.*sarg_peri;
P_hat(3,:)=sincl.*sarg_peri;
Q_hat(1,:)=-clan.*sarg_peri-slan.*cincl.*carg_peri;
Q_hat(2,:)=-slan.*sarg_peri+clan.*cincl.*carg_peri;
Q_hat(3,:)=sincl.*carg_peri;
%
rearthmars=zeros(6,size(t,2));
vearthmars=rearthmars;
```

```
rearth=zeros(3,size(t,2));
rmars=rearth;
vearth=rearth;
vmars=rearth;
%
rearth=(kmperAU*a_AU(1)*((cos(E(:,1))-ecc(:,1))*P_hat(:,1)' +sqrt(1-ecc(:,1).^2)
.*sin(E(:,1))*Q_hat(:,1)'))';
rmars=(kmperAU*a_AU(2)*((cos(E(:,2))-ecc(:,2))*P_hat(:,2)' +sqrt(1-ecc(:,2).^2)
.*sin(E(:,2))*Q_hat(:,2)'))';
vearth=(2*pi*kmperAU^2*a_AU(1)^2./Tp_yrs(1)/daysperyear/24/3600* (-
sin(E(:,1))./r(:,1)*P_hat(:,1)'+sqrt(1-eccentricity(1)^2) *cos(E(:,1))./r(:,1)*Q_hat(:,1)'))';
vmars=(2*pi*kmperAU^2*a_AU(2)^2./Tp_yrs(2)/daysperyear/24/3600* (-
sin(E(:,2))./r(:,2)*P_hat(:,2)'+sqrt(1-eccentricity(2)^2) *cos(E(:,2))./r(:,2)*Q_hat(:,2)'))';
```

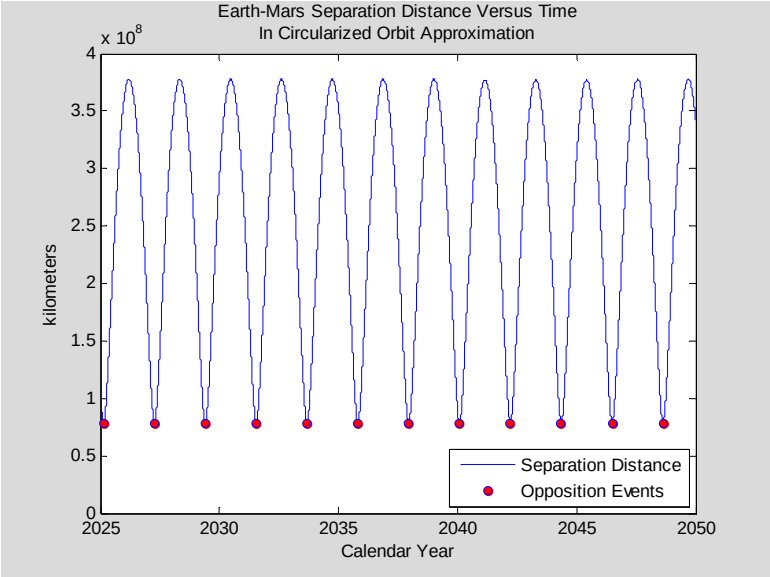
This is tested by the following code:

```
clear;clf;
t=linspace(datenum('1-Jan-2025'),datenum('31-dec-2030'),1000);
[rearth,rmars,vearth,vmars]=Circ2D_EarthMars(t);plot(0,0,'m.',rearth(1,
:),rearth(2,:),'b',rmars(1,:),rmars(2,:),'r'),ylim([-3e8
3e8]),xlabel('X (km)'),ylabel('Y (km)'),title({'Circularized 2D Earth &
Mars Approximate Orbits'  'Both In Ecliptic Plane'}),
legend('Sun','Earth','Mars','Location','SouthEast'), axis square
```



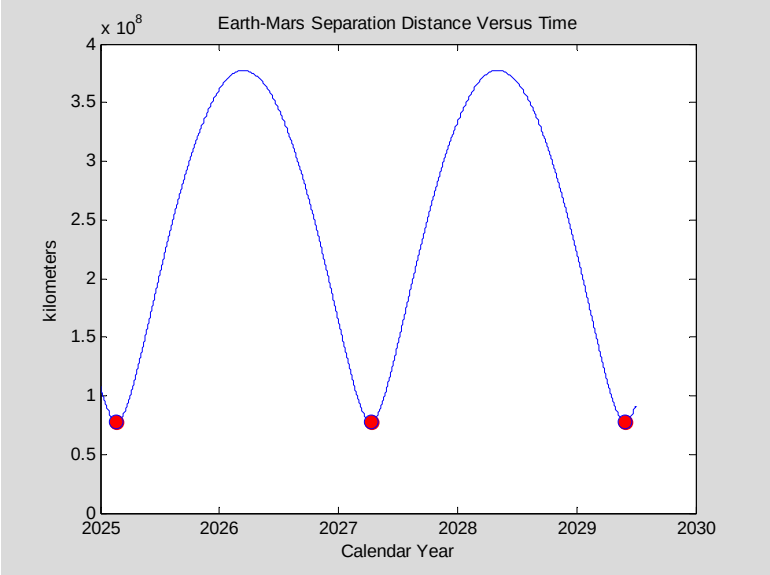

```
t=linspace(datenum('1-Jan-2025'),datenum('31-dec-2049'),10000);
[rearth,rmars,vearth,vmars]=Circ2D_EarthMars(t);
D_e_m=zeros(size(t));
for i=1:size(t,2);
D_e_m(1,i)=norm(rearth(:,i)-rmars(:,i));end,
III=find(diff(diff(D_e_m)<0)<0);Topposition=t(III);
```

```
plot(t',D_e_m',Topposition,D_e_m(III),'o','MarkerEdgeColor','b','Marker
FaceColor','r','MarkerSize',5),datetick('x'),xlabel('Calendar
Year'),ylabel('kilometers'),title({'Earth-Mars Separation Distance
Versus Time' 'In Circularized Orbit Approximation'}),ylim([0 4e8]),
datestr(Topposition),
legend('Separation Distance','Opposition
Events','Location','SouthEast')

ans =
19-Feb-2025 07:21:58
10-Apr-2027 02:05:08
28-May-2029 20:48:19
17-Jul-2031 15:31:29
04-Sep-2033 10:14:39
25-Oct-2035 02:52:40
12-Dec-2037 21:35:50
31-Jan-2040 16:19:00
21-Mar-2042 11:02:10
09-May-2044 05:45:20
28-Jun-2046 00:28:30
16-Aug-2048 17:06:32
```



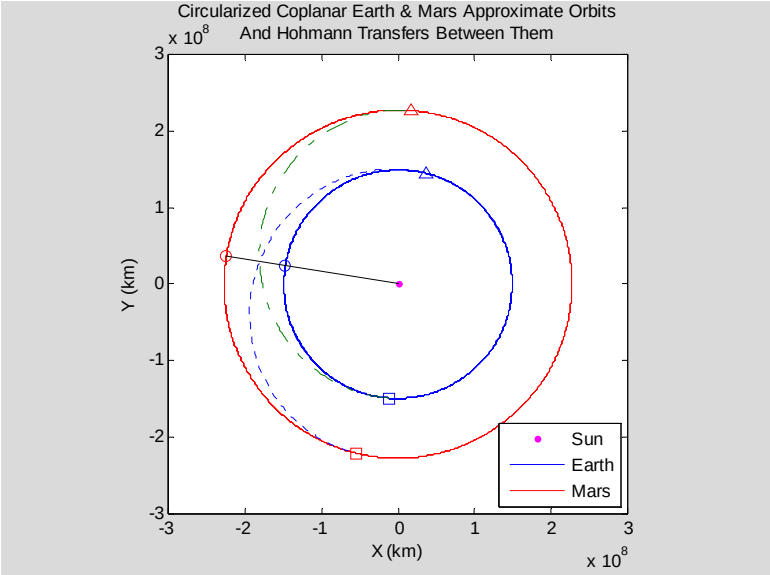

```
t=linspace(datenum('01-Jan-2025'),datenum('30-jun-2029'),10000);
[rearth,rmars,vearth,vmars]=Circ2D_EarthMars(t);
D_e_m=zeros(size(t));
for i=1:size(t,2);
D_e_m(1,i)=norm(rearth(:,i)-rmars(:,i));end,
III=find(diff(diff(D_e_m)<0)<0);Topposition=t(III);
plot(t',D_e_m',Topposition,D_e_m(III),'o','MarkerEdgeColor','b','Marker
FaceColor','r','MarkerSize',8),datetick('x'),xlabel('Calendar
Year'),ylabel('kilometers'),title('Earth-Mars Separation Distance
Versus Time'),ylim([0 4e8]),
datestr(Topposition)
```

```
ans =
19-Feb-2025 21:23:36
10-Apr-2027 22:28:50
29-May-2029 19:37:44
```




```
[rearth_xfer,rmars_xfer,vearth_xfer,vmars_xfer]=Circ2D_EarthMars([t(III
(2))-162.79 t(III(2))-96.07 t(III(2))+96.07 t(III(2))+162.79]);

re=norm(rearth_xfer(:,1));rm=norm(rmars_xfer(:,1));a=(re+rm)/2,eccentri
city=(rm-re)/(rm+re),
theta0_em=atan2(rearth_xfer(2,2),rearth_xfer(1,2)),
theta0_me=atan2(rearth_xfer(2,3),rearth_xfer(1,3))

rem_xfer=zeros(2,180);rme_xfer=rem_xfer;

theta=linspace(0,pi,180);

rem_xfer(1,:)=cos(theta+theta0_em)*a*(1-eccentricity^2)./
(1+eccentricity*cos(theta));
rem_xfer(2,:)=sin(theta+theta0_em)*a*(1-eccentricity^2)./
(1+eccentricity*cos(theta));

rme_xfer(1,:)=cos(-theta+theta0_me)*a*(1-eccentricity^2)./
(1+eccentricity*cos(-theta));
rme_xfer(2,:)=sin(-theta+theta0_me)*a*(1-eccentricity^2)./
(1+eccentricity*cos(-theta));

plot( 0,0,'m.',rearth(1,:)',rearth(2,:)','b',
rmars(1,:)',rmars(2,:)','r',
rem_xfer(1,:)',rem_xfer(2,:)',':',rme_xfer(1,:)',rme_xfer(2,:)','-.',
rearth(1,III(2)),rearth(2,III(2)),'ob',
rmars(1,III(2)),rmars(2,III(2)),'or',
rearth_xfer(1,2),rearth_xfer(2,2),'b^',
rearth_xfer(1,3),rearth_xfer(2,3),'bs',
```

```
rmars_xfer(1,1),rmars_xfer(2,1),'r^',
rmars_xfer(1,4),rmars_xfer(2,4),'rs',
[0;rmars(1,III(2))],[0;rmars(2,III(2))],'-k')

ylim([-3e8 3e8]),xlabel('X (km)'),ylabel('Y (km)'),title({'Circularized
Coplanar Earth & Mars Approximate Orbits' 'And Hohmann Transfers
Between Them'}), legend('Sun','Earth','Mars','Location','SouthEast'),
axis square

a =
  1.8877e+008
eccentricity =
    0.2075
theta0_em =
    1.3278
theta0_me =
   -1.6502
```



```
DeltaV_Rocket=[];DeltaV_Reentry=[];
tbegin=datenum('01-Jan-2024');
tend=datenum('01-Jan-2031');
Tearth=[];
ttransfer=([-365:5:-30 0 30:5:365])';
mdata=size(ttransfer,1);
%
while isempty(Tearth) | Tearth(end)<tend;
deltav_rocket=zeros(mdata,1);deltav_reentry=zeros(mdata,1);
if isempty(Tearth)
 Tearth=tbegin;
else
 Tearth=[Tearth (Tearth(end)+5)];
end
```

```
tearth=Tearth(end);
for i=1:mdata;
 if ttransfer(i)<0;
[deltav_rocket(i),deltav_reentry(i)]=Circ2D_Mars2Earth(ttransfer(i)+tea
rth,tearth);
 elseif ttransfer(i)>0;
[deltav_rocket(i),deltav_reentry(i)]=Circ2D_Earth2Mars(tearth,tearth+tt
ransfer(i));
else
  deltav_rocket(i)=NaN; deltav_reentry(i)=NaN;
end
end
DeltaV_Rocket=[DeltaV_Rocket deltav_rocket];
DeltaV_Reentry=[DeltaV_Reentry deltav_reentry];
end
'done'
beep
```

```
ans =
done
```

```
save Circ2D_OrbitalTransferData
```

```
load Circ2D_OrbitalTransferData
```

```
[m,n]=size(DeltaV_Rocket),
[II,JJ]=find([zeros(m,1) (diff((diff(DeltaV_Rocket,1,2)<0),1,2)<0)
zeros(m,1)] & [zeros(1,n); (diff((diff(DeltaV_Rocket,1,1)<0),1,1)<0) ;
zeros(1,n)]);Indices=[II JJ]
disp(['Earth         Mars      DeltaV Rocket']),
disp([datestr(Tearth(Indices(:,2))) blanks(54)' blanks(54)'
datestr(Tearth(Indices(:,2))+ttransfer(Indices(:,1))') blanks(54)'
blanks(54)' num2str(diag(DeltaV_Rocket(Indices(:,1),Indices(:,2)))) ])
```

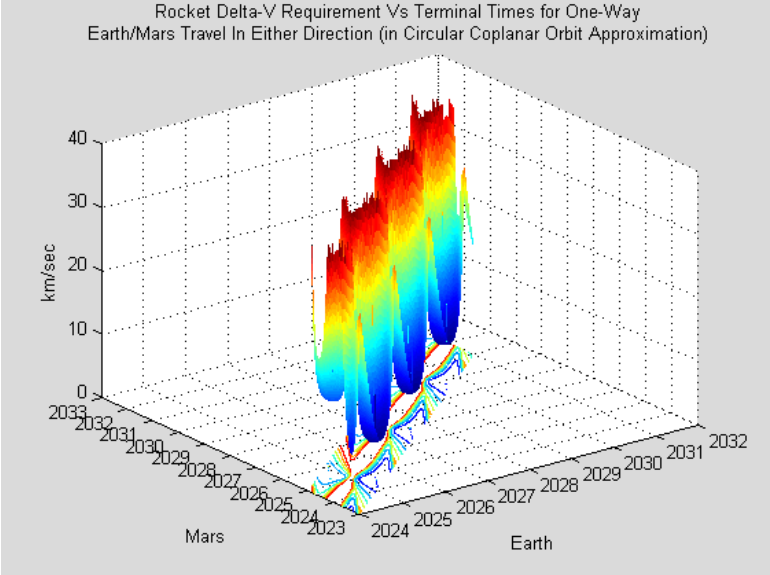
```
m =
   137
n =
   513
Indices =
   115    61
   116    62
   117    63
   118    64
   115    65
   119    65
   120    66
   118    67
   119    68
   120    69
   123    73
   124    74
```

```
   125    75
   135    87
   136    88
    23   102
    22   103
    21   105
   115   217
   116   218
   117   219
   118   220
   115   221
   119   221
   117   222
   120   222
   120   225
   121   226
   124   230
   125   231
   126   232
   129   236
   130   237
   131   238
   135   243
   136   244
    23   259
    21   261
   116   374
   118   376
   115   377
   119   377
   120   378
   120   381
   124   386
   125   387
   126   388
   130   393
   131   394
   135   399
   136   400
    23   415
    22   416
    21   417
Earth          Mars      DeltaV Rocket
27-Oct-2024  09-Jul-2025   11.654
01-Nov-2024  19-Jul-2025  11.6156
06-Nov-2024  29-Jul-2025  11.5888
11-Nov-2024  08-Aug-2025  11.5726
16-Nov-2024  29-Jul-2025  11.5611
16-Nov-2024  18-Aug-2025  11.5658
21-Nov-2024  28-Aug-2025  11.5676
26-Nov-2024  23-Aug-2025  11.5804
01-Dec-2024  02-Sep-2025  11.6035
06-Dec-2024  12-Sep-2025  11.6354
26-Dec-2024  17-Oct-2025  11.8248
31-Dec-2024  27-Oct-2025  11.8833
```

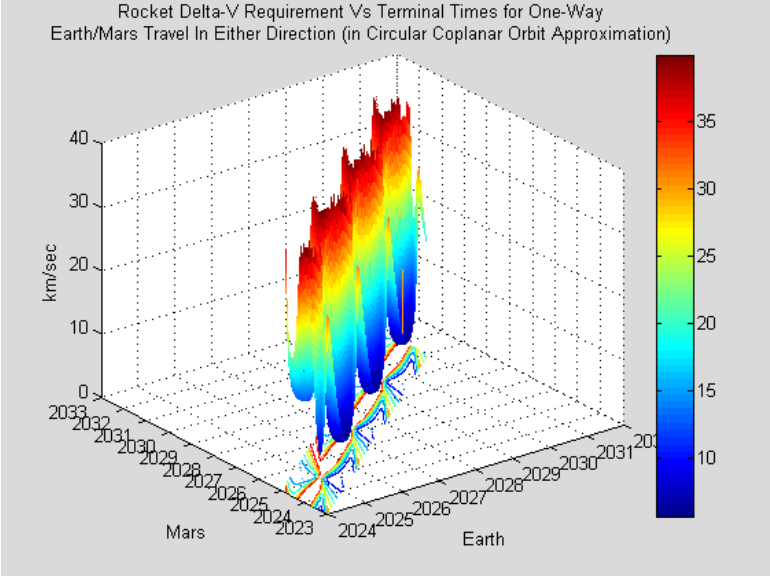
```
05-Jan-2025  06-Nov-2025   11.945
06-Mar-2025  24-Feb-2026  12.9479
11-Mar-2025  06-Mar-2026  13.0269
20-May-2025  07-Sep-2024   5.7031
25-May-2025  07-Sep-2024   5.7016
04-Jun-2025  12-Sep-2024  5.70776
16-Dec-2026  28-Aug-2027  11.6532
21-Dec-2026  07-Sep-2027  11.6149
26-Dec-2026  17-Sep-2027  11.5893
31-Dec-2026  27-Sep-2027  11.5722
05-Jan-2027  17-Sep-2027   11.561
05-Jan-2027  07-Oct-2027  11.5657
10-Jan-2027  02-Oct-2027  11.5652
10-Jan-2027  17-Oct-2027  11.5676
25-Jan-2027  01-Nov-2027   11.631
30-Jan-2027  11-Nov-2027  11.6678
19-Feb-2027  16-Dec-2027  11.8844
24-Feb-2027  26-Dec-2027  11.9464
01-Mar-2027  05-Jan-2028  12.0119
21-Mar-2027  09-Feb-2028  12.3353
26-Mar-2027  19-Feb-2028  12.4113
31-Mar-2027  29-Feb-2028  12.4881
25-Apr-2027  14-Apr-2028  12.9508
30-Apr-2027  24-Apr-2028    13.03
14-Jul-2027  01-Nov-2026  5.70029
24-Jul-2027  01-Nov-2026  5.70624
08-Feb-2029  26-Oct-2029  11.6147
18-Feb-2029  15-Nov-2029  11.5721
23-Feb-2029  05-Nov-2029   11.561
23-Feb-2029  25-Nov-2029  11.5656
28-Feb-2029  05-Dec-2029  11.5676
15-Mar-2029  20-Dec-2029  11.6336
09-Apr-2029  03-Feb-2030  11.8873
14-Apr-2029  13-Feb-2030  11.9507
19-Apr-2029  23-Feb-2030  12.0193
14-May-2029  09-Apr-2030  12.4138
19-May-2029  19-Apr-2030  12.4909
13-Jun-2029  03-Jun-2030  12.9534
18-Jun-2029  13-Jun-2030  13.0326
01-Sep-2029  20-Dec-2028  5.70029
06-Sep-2029  20-Dec-2028  5.70042
11-Sep-2029  20-Dec-2028  5.70379
```

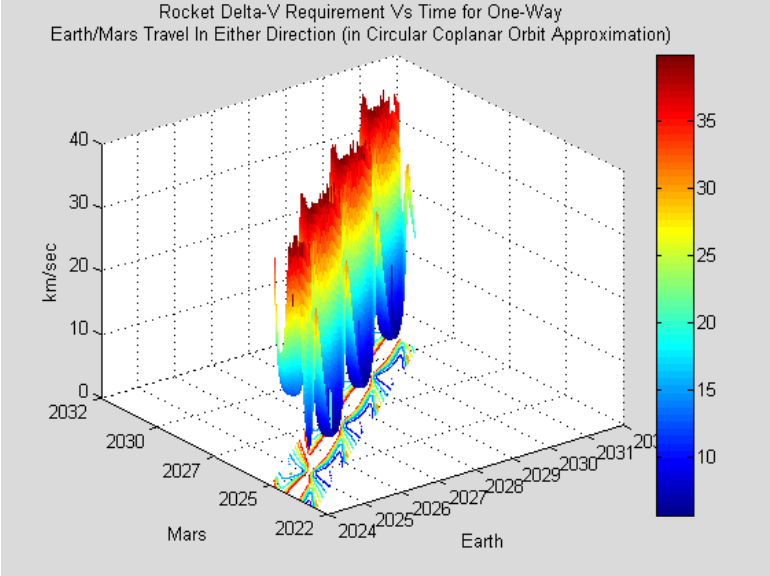
```
ClippedDeltaV_Rocket=DeltaV_Rocket;
for i=1:137;
for j=1:513;
if ClippedDeltaV_Rocket(i,j)>40;
ClippedDeltaV_Rocket(i,j)=NaN;
end
end
end
```

```
surfc(XXX,YYY,ClippedDeltaV_Rocket,'EdgeColor','none');
datetick('x');datetick('y');
zlabel('km/sec');
xlabel('Earth');
ylabel('Mars');
title({'Rocket Delta-V Requirement Vs Terminal Times for One-Way'
'Earth/Mars Travel In Either Direction (in Circular Coplanar Orbit
Approximation)'});
```

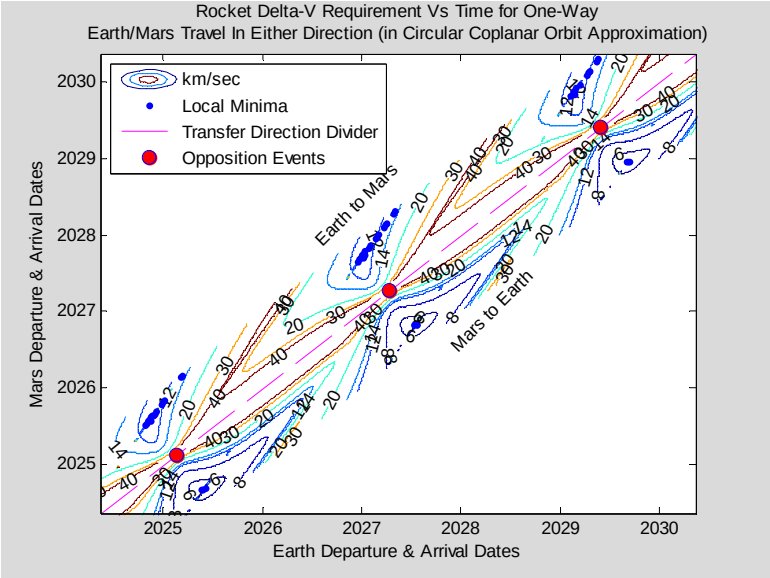



```
colorbar
```

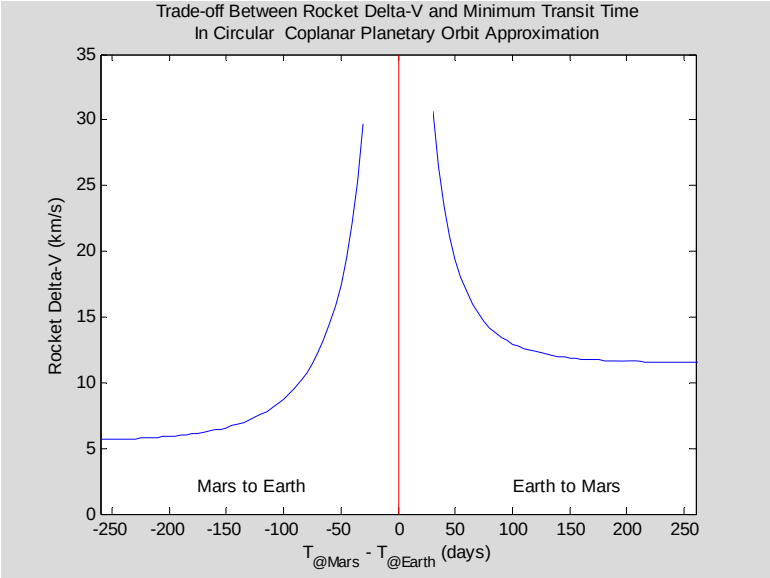





```
clf;
[XXX,YYY]=meshgrid(Tearth,ttransfer);
YYY=YYY+XXX;
tmax=max(max(max(XXX)'),max(max(YYY)'));
tmin=min(min(min(XXX)'),min(min(YYY)'));
[CC,hh]=contour(XXX,YYY,DeltaV_Rocket,[6 8 12 14 20 30 40]);
datetick('x'),datetick('y'),
xlim([tmin+500 tmax-600]);ylim([tmin+500 tmax-600]);
tmid=(tmin+tmax)/2;
clabel(CC,hh);
xlabel('Earth Departure & Arrival Dates'), ylabel('Mars Departure &
Arrival Dates'),
title({'Rocket Delta-V Requirement Vs Time for One-Way'  'Earth/Mars
Travel In Either Direction (in Circular Coplanar Orbit
Approximation)'})
hold on;
plot(Tearth(Indices(:,2)),Tearth(Indices(:,2))+ttransfer(Indices(:,1))'
,'.',[tmin tmax],[tmin tmax], '--m',
Topposition,Topposition,'o','MarkerEdgeColor','b','MarkerFaceColor','r'
,'MarkerSize',8),legend('km/sec','Local Minima','Transfer Direction
Divider','Opposition Events','Location','NorthWest');hold
off;text(tmid-100-250,tmid-100+250,'Earth to
Mars','Rotation',45);text(tmid-100+250,tmid-100-250,'Mars to
Earth','Rotation',45)
```

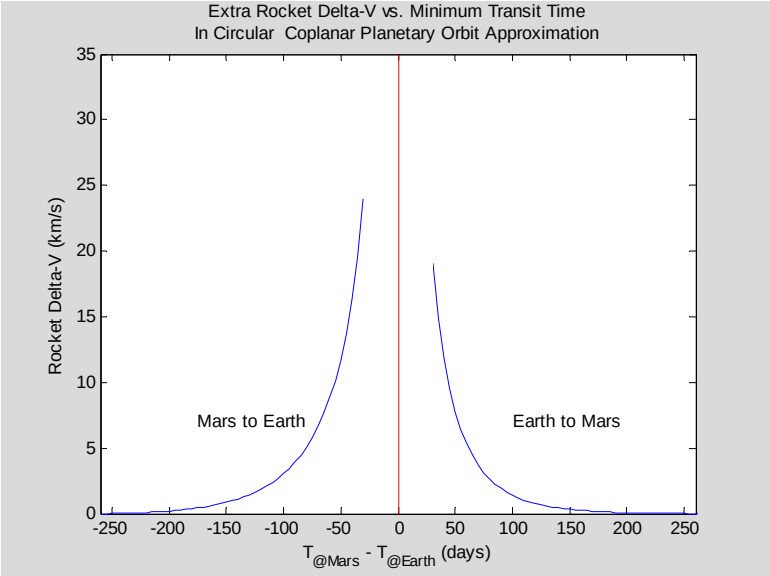



```
DeltaV_Tradeoff=min(DeltaV_Rocket')';
plot([0 0],[0 35],'-r',ttransfer,DeltaV_Tradeoff,'-b');
ylim([0 35]);
ylabel('Rocket Delta-V (km/s)')
xlim([-260 260]),
xlabel('T_@_M_a_r_s - T_@_E_a_r_t_h (days)')
title({'Trade-off Between Rocket Delta-V and Minimum Transit Time' 'In
Circular  Coplanar Planetary Orbit Approximation'}),
text(-175,2,'Mars to Earth'),text(+100,2,'Earth to Mars')
```

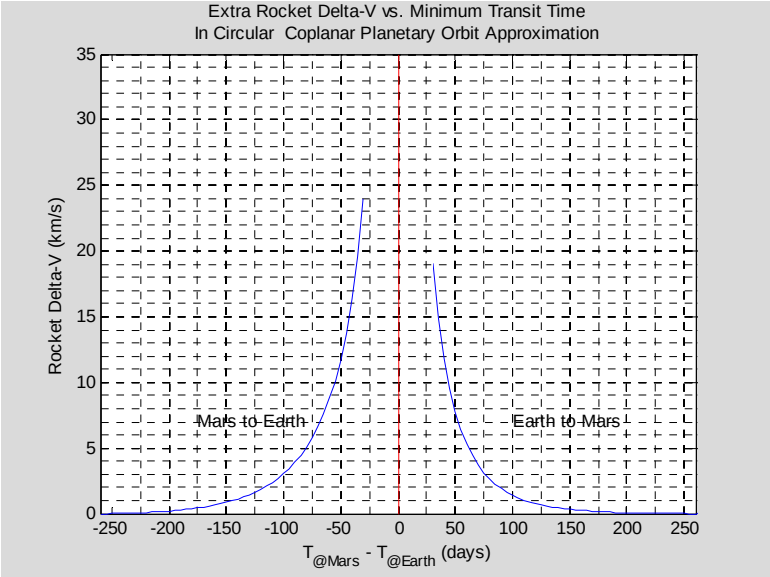



```
DeltaV_Extra=DeltaV_Tradeoff;
DeltaV_Extra(1:68)=DeltaV_Extra(1:68)-DeltaV_Tradeoff(22);
DeltaV_Extra(70:end)=DeltaV_Extra(70:end)-DeltaV_Tradeoff(116);
```

```
plot([0 0],[0 35],'-r',ttransfer,DeltaV_Extra,'-b');
ylim([0 35]);
ylabel('Rocket Delta-V (km/s)')
xlim([-260 260]),
xlabel('T_@_M_a_r_s - T_@_E_a_r_t_h (days)')
title({'Extra Rocket Delta-V vs. Minimum Transit Time' 'In Circular
Coplanar Planetary Orbit Approximation'}),
text(-175,7,'Mars to Earth'),text(+100,7,'Earth to Mars')
```

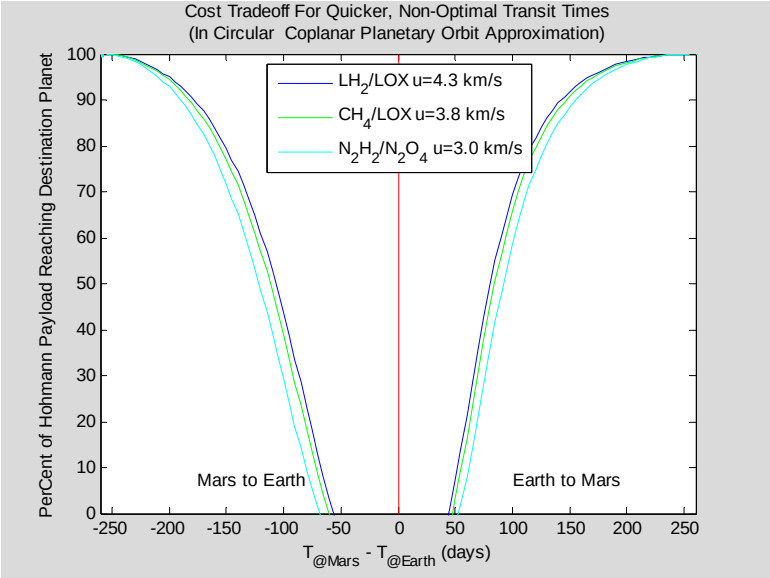



```
grid minor
```

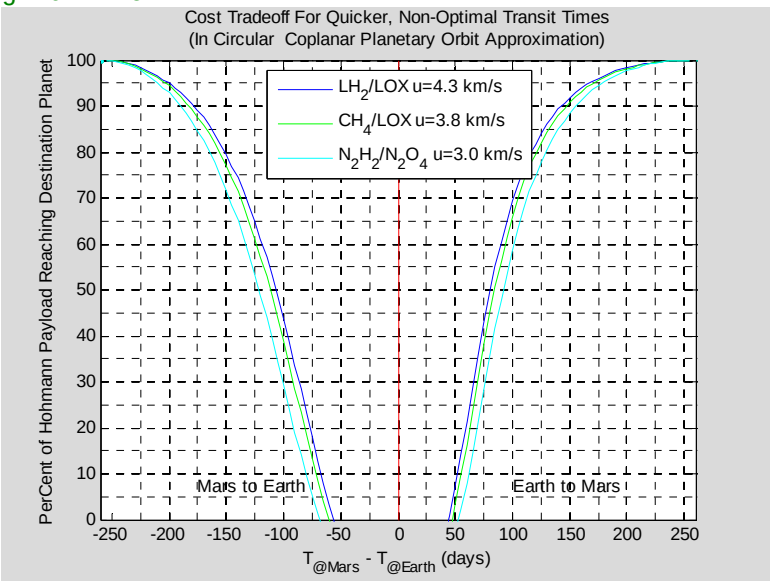



```
PayloadFraction4300=(10*exp(-DeltaV_Extra/4.3)-1)/9.;
PayloadFraction3800=(10*exp(-DeltaV_Extra/3.8)-1)/9.;
PayloadFraction3000=(10*exp(-DeltaV_Extra/3.0)-1)/9.;
```

```
plot(ttransfer,100*PayloadFraction4300,'b',
ttransfer,100*PayloadFraction3800,'g',
ttransfer,100*PayloadFraction3000,'c', [0 0],[0 100],'-r');
legend('LH_2/LOX u=4.3 km/s','CH_4/LOX u=3.8 km/s','N_2H_2/N_2O_4 u=3.0
km/s','Location','North')
ylim([0 100]);
ylabel('PerCent of Hohmann Payload Reaching Destination Planet')
xlim([-260 260]),
xlabel('T_@_M_a_r_s - T_@_E_a_r_t_h (days)')
title({'Cost Tradeoff For Quicker, Non-Optimal Transit Times' '(In
Circular  Coplanar Planetary Orbit Approximation)'}),
text(-175,7,'Mars to Earth'),text(+100,7,'Earth to Mars')
```




```
grid minor
```



APPENDIX C
C MATLAB MODELS OF FLUID PROPERTIES

C.1 Summary

This appendix is configured as a MATLAB/MSWORD Notebook in order to internally implement computations to investigate characteristics of fluids relevant to the present thesis project. Formatting within this appendix follows MATLAB/MSWORD Notebook norms since they contain specially formatted “cells” of MATLAB code written in a particular special font and delineated by special hidden characters. MATLAB/MSWORD Notebook documents also contain output results from the specified computations, which may be graphical plots or numerical tables using a different special font. Such documents deviate from conventions for easily readable material since they are essentially executable computer codes and their resulting outputs along with some interspersed explanatory material. For this thesis, they allow the reader to perform a detailed audit of calculations made and they also make it possible for the reader to redo or extend the calculations by use of the “cut and paste” function.

The mobile nuclear engine designs investigated in this work rely on fluids for convective heat transfer, for thermal energy conversion, and for waste heat rejection. Relevant properties of these fluids must therefore be modeled with sufficient fidelity to support analyses. To that end, fluid property data were acquired from published sources, then approximate algorithms fitted to the properties data were synthesized and implemented in MATLAB code which is embedded in this appendix and which has the combined executable file format of a MATLAB notebook/MSWORD m-book. (The embedded code may be re-executed by any "desktop" computer on which both Microsoft Word software and MATLAB software along with MATLAB's optional "spline toolbox" has been installed.) This appendix presents and discusses these fluid property algorithms and compares them with the published data which they approximate.

The open cycle nuclear reactor engine designs which are the main focus of this work use molten lithium metal as their primary reactor coolant heat transfer fluid and use martian air both for thermal energy conversion and for waste heat rejection. Thermal-hydraulic properties are approximated for the molten lithium, which is treated as an incompressible liquid with temperature-dependent density. The martian air is approximated as a fixed-proportion mixture (herein dubbed "MarsMix") of the three main gaseous components comprising the martian atmosphere, i.e., carbon dioxide, argon, and nitrogen, with each component modeled as a compressible perfect gas with variable temperature-dependent specific heats. Properties of this MarsMix approximant to Martian air are calculated here as molar weighted averages of the properties of the three component gases. Since this work also briefly compares performance with engines using "Earth-air" or helium, relevant properties of these gases are also addressed in this appendix. Fluid properties are modeled herein as per Table C-1.

Table C-1: Variable Fluid Properties Modeled in MATLAB

Property\Fluid	Gases	Liquid Lithium
Density	**IdealGasLaw (ρ=p/(RT)**	**Nonlinear Function ofTemperature**
Enthalpy	**Nonlinear Function ofTemperature**	**Nonlinear Function ofTemperature**
Entropy	**Nonlinear Function ofTemperature**	**(not modeled)**
Specific Heat	**Nonlinear Function ofTemperature**	**Nonlinear Function ofTemperature**
Thermal Conductivity	**Nonlinear Function ofTemperature**	**Nonlinear Function ofTemperature**
Viscosity	**Nonlinear Function ofTemperature**	**Nonlinear Function ofTemperature**

Fluid properties are approximated here based on least-squares fitting of piecewise cubic "spline" functions to the published data, as discussed in detail herein. The resulting temperature-dependent fluid property algorithms have been implemented in MATLAB m-file subroutine form to facilitate analyses. These fluid property algorithms accurately match the data up to 1500 Kelvins and may be sufficiently accurate for the calculations of this work up to 2000 Kelvins. Some of the fluid properties may be accurate up to considerably higher temperatures while others are not.

C.2 Essential Theory

For Brayton cycle analyses, the working fluid is modeled here as a *perfect gas,* [Reynolds 1965, 220-231], which is any gas obeying the following gas law:

$$Pv = RT \tag{C-1}$$

where
P is the gas pressure (Pascals)
T is the gas temperature (Kelvins), and
R is the gas constant for this composition of gas, (joules/kg/Kelvin)
v is the specific volume of the gas, (m^3/kg),
defined as the inverse of density, i.e., $\rho = \frac{1}{v}$ where ρ is the gas density (kg/m^3). All gases with sufficiently low densities obey Eq. (1). The gases considered in Brayton cycle analyses herein obey Eq. (1) quite accurately at the densities encountered.

The gas constant, R, is related to the *molal mass*, $\hat{M}$ (kg/mole), and to the *universal gas constant*, $\mathcal{R}$, as:

$$R = \frac{\mathcal{R}}{\hat{M}} \tag{C-2}$$

where the value of the universal gas constant in SI units is 8.31361 J/mole/Kelvin.

For mixtures of perfect gases, Eq.(1) applies for each mixture component but with pressure replaced by the component's partial pressure and with specific volume replaced by component's specific volume. Component partial pressures and specific volumes are each proportional to the component's mole fraction.

Although for any simple compressible substance the internal energy, u, is generatlly expressed as a function of temperature and specific volume, i.e., $u = u(T, v)$, it can be shown that Eq. (1) implies that there is no dependence of internal energy on specific volume. Thus, for a perfect gas, internal energy depends only on gas temperature.

$$u = u(T) \tag{C-3}$$

It follows from Eq. (3) that the differential of internal energy is the specific heat at constant volume times the differential of temperature,

$$du = c_v dT \tag{C-4}$$

and that the specific heat at constant volume itself also only depends on temperature.

$$c_v = c_v(T) \tag{C-5}$$

It also follows that enthalpy is only a function of temperature, as

$$h \equiv u + Pv = u(T) + RT = h(T) \tag{C-6}$$

Thus for a perfect gas the differential of enthalpy is simply the specific heat at constant pressure times the temperature differential,

$$dh \equiv c_P dT \tag{C-7}$$

and the specific heat at constant pressure is also only dependent on temperature.

$$c_P = c_P(T) \tag{C-8}$$

Equations (4), (6), and (7) together show that for a perfect gas,

$$c_P = c_v + R \tag{C-9}$$

Entropy of a perfect gas is equivalently defined either by the Gibbs differential equation,

$$ds = \frac{du}{T} + \frac{P}{T} dv = \frac{c_v}{T} dT + R \frac{dv}{v} \tag{C-10}$$

or by the following differential equation:

$$ds = \frac{dh}{T} - \frac{v}{T} dP = \frac{c_P}{T} dT - R \frac{dP}{P} \tag{C-11}$$

Differential equations (4), (7) , (10), and (11) may be integrated over a temperature range to give the following perfect gas relations in integral form:

$$u_2 - u_1 = \int_{T_1}^{T_2} c_v(T) dT \tag{C-12}$$

$$h_2 - h_1 = \int_{T_1}^{T_2} c_P(T)dT \tag{C-13}$$

$$s_2 - s_1 = \int_{T_1}^{T_2} \frac{c_v(T)}{T} dT + R\ln\left(\frac{v_2}{v_1}\right) \tag{C-14}$$

$$s_2 - s_1 = \int_{T_1}^{T_2} \frac{c_P(T)}{T} dT - R\ln\left(\frac{P_2}{P_1}\right) \tag{C-15}$$

By invoking Eq. (9), it follows that Eqs.(12) and (14) can be rewritten equivalently as:

$$u_2 - u_1 = \int_{T_1}^{T_2} c_P(T)dT - R(T_2 - T_1) \tag{C-16}$$

and

$$s_2 - s_1 = \int_{T_1}^{T_2} \frac{c_P(T)}{T} dT + R\ln\left(\frac{v_2 T_1}{v_1 T_2}\right) \tag{C-17}$$

At this point, Eqs. (13), (15), (16), and (17) define the important thermodynamic properties in terms of the specific heat at constant pressure, $c_P(T)$, a function of temperature which has been tabulated for many gases based on empirical measurements.

For special cases in which the specific heat at constant pressure does not change much over a temperature range of interest, it can be sufficiently accurate to approximate it as a constant. For such constant specific heat cases the integrals in Eqs. (12), (15),(16),(17) can be evaluated in closed form, yielding the following:

$$u_2^{(\text{Constant Cp})} - u_1^{(\text{Constant Cp})} = (T_2 - T_1)c_v \tag{C-18a}$$

$$h_2^{(\text{Constant Cp})} - h_1^{(\text{Constant Cp})} = (T_2 - T_1)c_P \tag{C-18b}$$

$$s_2^{(\text{Constant Cp})} - s_1^{(\text{Constant Cp})} = c_v \ln\left(\frac{T_2}{T_1}\right) - R\ln\left(\frac{P_2}{P_1}\right) \tag{C-18c}$$

It is conventional when analyzing thermodynamics of engines employing gases with constant specific heats to define the *ratio of specific heats*, γ, as

$$\gamma \equiv \frac{c_P}{c_v} = \frac{c_P}{c_P - R} \tag{C-18d}$$

Substituting this and Eqs.(9) and (16) into Eq. (17c) yields, after algebraic operations, the following relation between any two states:

$$P_2 v_2^{\gamma} = P_1 v_1^{\gamma} \exp\left(\frac{s_2^{(\text{Constant Cp})} - s_1^{(\text{Constant Cp})}}{c_v} \right) \qquad \text{(C-!8e)}$$

This last relation, whose validity depends on its constant specific heat assumption, is used widely in technical literature to model isentropic processes such as adiabatic gas compression or expansion. A "polytropic" modification maintaining this form is also commonly used to model loss effects in more realistic analyses of nonideal compressors and turbines.

The constant specific heat approximation is excellent for a monotomic gas like helium and can be adequate for the diatomic gases such as nitrogen and oxygen which dominate Earth's atmosphere. However, it is a poor approximation for carbon dioxide, the gas which dominates Mars' atmosphere. (Viking project measurements showed Mars' time-averaged atmosphere composition expressed in volume fractions is 95.32% carbon dioxide, 2.70 % nitrogen, 1.60 % argon, 0.13% oxygen, 0.08% carbon monoxide (CO), 0.02% water vapor, 0.01% nitrogen oxide, with the remaining 0.14% divided between neon ,krypton, xenon and others). The specific heat at constant pressure of carbon dioxide varies by about 60% over the temperature range from 300 to 1300 Kelvins which is of most interest for Mars vehicle engines. With so much variation, attempts to model the predominately CO2 working fluid as having constant specific heats could cause large errors in calculated isentropic compression and expansion processes. Therefore, Eqs. (18) are <u>not</u> used in analyses herein of the open Brayton cycle using martian air.

Approximately modeling martian air as pure carbon dioxide might be sufficiently accurate for this study. However, it was decided to instead model the martian air as being a gas mixture with fixed volume fractions of 95.7% carbon dioxide, 2.70% nitrogen, and 1.60% argon. This gas mixture is herein dubbed "MarsMix". Inclusion of the small amounts of nitrogen and argon was adopted in an attempt to improve the approximation beyond that of pure carbon dioxide. This approximation does ignore the seasonal variation of carbon dioxide content as carbon dioxide gas sublimes from Mars' sunlit pole's ice cap and forms a fresh dry ice layer on the dark pole's ice cap. This bi-annual variation causes the nitrogen and argon volume fractions to oscillate slightly about their mean values. The MarsMix gas approximation also ignores other trace gas atmospheric constituents since they seem likely to have insignificant effects on engine performance.

By modeling the MarsMix gas as a non-reacting mixture of *independent substances*, each separately modeled as a perfect gas, it follows that the volume fraction of each component is equal to its mole fraction and also to its partial pressure fraction [Reynolds 1965, 302-9]. The mixture's *per mole* values of internal energy, enthalpy, entropy, specific heats at either constant pressure or constant volume, and molal mass, are then

volume-fraction-weighted averages of the corresponding per mole properties of each component.

Thus, the molal mass of MarsMix gas is

$$\begin{aligned}\hat{M}_{MarsMix} &= 0.957\hat{M}_{CO2} + 0.027\hat{M}_{N2} + 0.016\hat{M}_{Ar} \\ &= 0.957(0.0440098\ \text{kg/mole}) + 0.027(0.0280134\ \text{kg/mole}) + 0.016(0.039948\ \text{kg/mole}) \\ &\approx 0.043513\ \text{kg/mole}\end{aligned} \tag{C-19}$$

The gas constant to use for MarsMix in Eq. (1) is then

$$\begin{aligned}\mathrm{R} &= \frac{\mathcal{R}}{\hat{M}_{MarsMix}} = \frac{8.31361\ \text{J/mole/Kelvin}}{0.043513\ \text{kg/mole}} \\ &= 191.06\ \text{J Kelvin}^{-1}\ \text{kg}^{-1}\end{aligned} \tag{C-20}$$

Some of the temperature-dependent gas properties (including specific heat, enthalpy,entropy) for the CO2, Argon, and Nitrogen components were taken from the NIST-JANAF Thermochemical Tables [JANAF which historically were compiled at the National Bureau of Standards under Joint Army-Navy-Air Force funding. Temperatures in the tables span the range from 0 to 6000 Kelvins. The specific heat (Cp) and the entropy integral (S0) are listed in Joules per Kelvin per mole, while the enthalpy, H0, is listed in kilojoules per mole *deviation* from the enthalpy reference at 25C. The tabulated entropy integral's lower limit is absolute zero, i.e.,

$$S0(T) \equiv \int_0^T \frac{c_P(T)}{T}\,dT \tag{C-21}$$

Although the simple algebraic relation of Eq. (18e) cannot be used for variable specific heat gases, Eq. (21) can be invoked to yield a valid numerical functional relation involving the tabulated functions.

$$\frac{P_2}{P_1} = \exp\left(+\frac{S0(T_2) - S0(T_1) - (s_2 - s_1)}{R}\right) \tag{C-22a}$$

and

$$\frac{v_2}{v_1} = \frac{T_2}{T_1}\exp\left(-\frac{S0(T_2) - S0(T_1) - (s_2 - s_1)}{R}\right) \tag{C-22b}$$

Tabulated properties at absolute zero do not appear meaningful, so interpolated properties at any temperature below the next tabulated data point at100 Kelvins may perhaps not be accurate. Happily, such cryogenic temperatures are not needed for analyses of proposed

air-breathing engines for Mars, since although Mars is frigid, it never gets down to 100 Kelvins anywhere. (Indeed, at typical martian air pressure the carbon dioxide comprising most of Mars' atmosphere freezes out as solid dry ice at temperatures near 145 Kelvins.)

Other important gas properties include the viscosity and the thermal conductivity. Although these are not included in the JANAF tables, they are tabulated in another reference book, Tables of Thermal Properties of Gases, National Bureauof Standards Circular 564, November 1955, US Government Printing Office, by Hilsenrath, Beckett,Benedict, Fano, Hoge, Masi, Nuttall, Touloukian,Woolley. For the component gases, Argon, Nitrogen, and Carbon Dioxide, the viscosity and the thermal conductivity at 1 atmosphere pressure are separately tabulated as functions of temperature. The tables have been copied for use here.

In addition to these data sources, other references consulted included the CRC Handbook of Chemistry and Physics, Eckert's text on heat transfer, some journal publications, and internet listings. Data extracted from them are also displayed here.

Specific heat and enthalpy are separately modeled herein because they are available separately in consulted data sources. In principle, one could be obtained from the other since as modeled herein the specific heat at constant pressure should be equal to the derivative with respect to temperature of the enthalpy.

C.3 Argon Gas Properties

C.3.1 Argon Gas Density

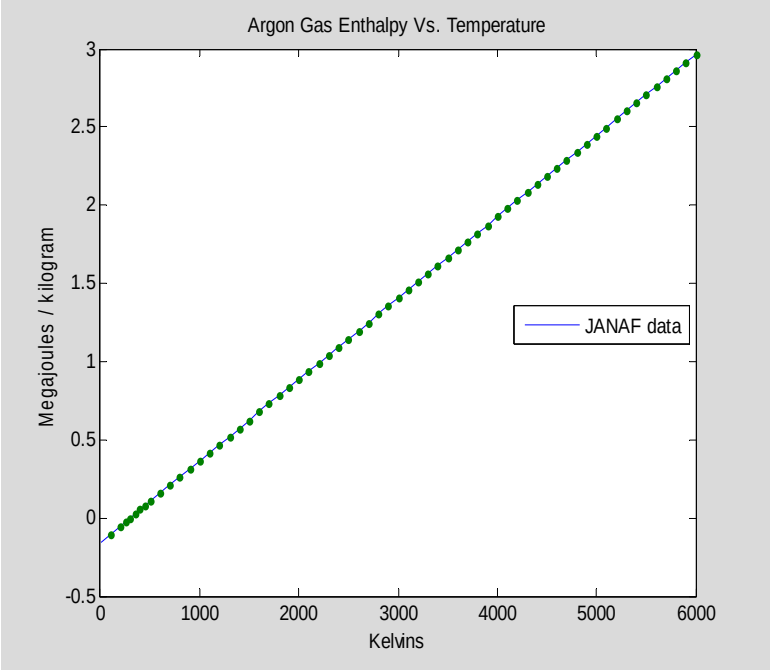
$$\rho = \frac{p}{RT} \text{ where } R = \frac{\mathcal{R}}{\hat{M}} = \frac{8.31361 \text{ Joule/mole/Kelvin}}{0.039948 \text{ kg/mole}} = 208.1108 \text{ Joule kg}^{-1} \text{ Kelvin}^{-1}$$

C.3.2 Argon Gas Enthalpy

Argon enthalpy is taken from the published JANAF tables as follows. Enthalpy values here are given in kilojoules per mole and are tabulated vs temperatures in Kelvins. The mass of a mole of argon is 0.039948 kilogram, a fact used to convert enthalpy to the megajoules per kilogram form plotted in the graph below.

```
TK_H0_Ar=[0;100;200;250;298.15;300;350;400;450;500;600;700;800;900;1000
;1100;1200;1300;1400;1500;1600;1700;1800;1900;2000;2100;2200;2300;2400;
2500;2600;2700;2800;2900;3000;3100;3200;3300;3400;3500;3600;3700;3800;
3900;4000;4100;4200;4300;4400;4500;4600;4700;4800;4900;5000;5100;5200;
5300;5400;5500;5600;5700;5800;5900;6000];
H0_Ar =[-6.197;-4.119;-2.04;-1.001;0;0.038;1.078;2.117;3.156;4.196;
6.274;8.353;10.431;12.51;14.589;16.667;18.746;20.824;22.903;24.982;27.0
6;
29.139;31.217;33.296;35.375;37.453;39.532;41.61;43.689;45.768;47.846;
```

```
49.925;52.004;54.082;56.161;58.239;60.318;62.397;64.475;66.554;68.632;
70.711;72.79;74.868;76.947;79.025;81.104;83.183;85.261;87.34;89.418;
91.497;93.576;95.654;97.733;99.811;101.89;103.969;106.047;108.126;
110.204;112.283;114.362;116.44;118.519];
plot(TK_H0_Ar,H0_Ar/0.039948/1000,TK_H0_Ar,H0_Ar/0.039948/1000,'.'),
xlabel('Kelvins'),
ylabel('Megajoules / kilogram'),
title('Argon Gas Enthalpy Vs. Temperature'),
legend('JANAF data','Location','East')
```

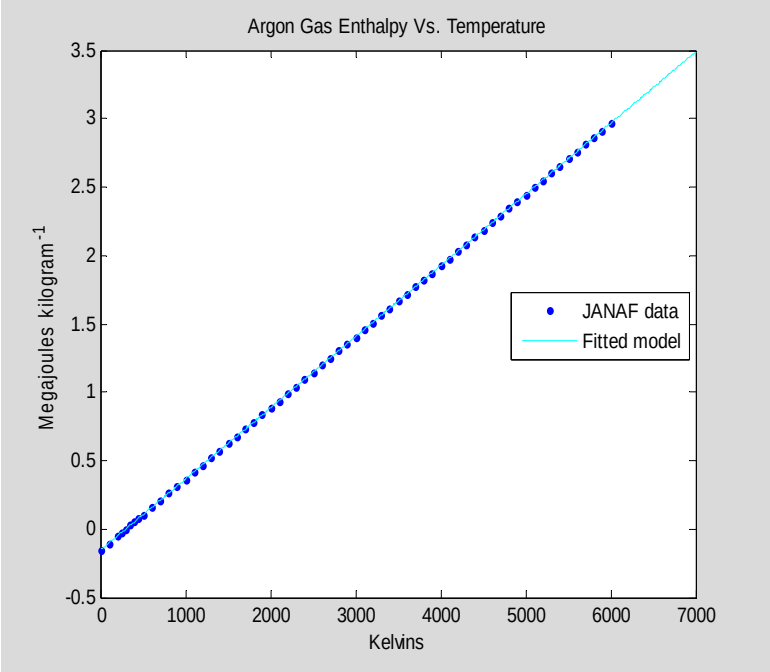


Next we approximate this data by using least squares techniques to fit cubic splines with a priori specified knot sequences.

```
knots=augknt([0 100 500 1000 1500 2000 3000 4000 5000 6000],4);
H0_Ar_pp=fn2fm(spap2(knots,4,TK_H0_Ar,H0_Ar/0.039948*1000),'pp');
H0_Ar_fit=fnval(H0_Ar_pp,linspace(000,7000,500));
plot(TK_H0_Ar,H0_Ar/0.039948/1000,'.',linspace(000,7000,500),
H0_Ar_fit/1e6,'-c'), xlabel('Kelvins'), ylabel('Megajoules kilogram^-
^1'), title('Argon Gas Enthalpy Vs. Temperature'), legend('JANAF
data','Fitted model','Location','East'),
format long g,
display(H0_Ar_pp),
H0_Ar_pp.coefs(:,1),
H0_Ar_pp.coefs(:,2),
H0_Ar_pp.coefs(:,3),
H0_Ar_pp.coefs(:,4)
```

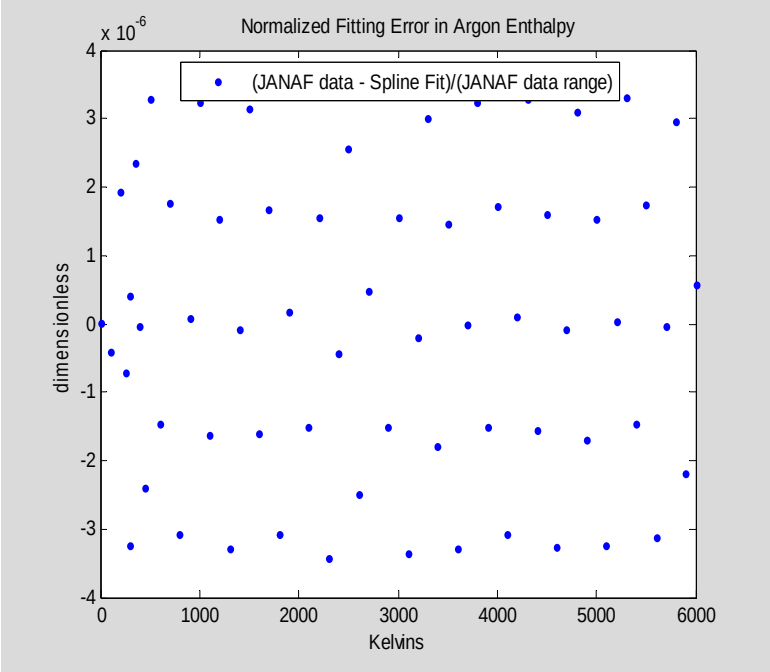
```
H0_Ar_pp =
      form: 'pp'
    breaks: [0 100 500 1000 1500 2000 3000 4000 5000 6000]
```

```
     coefs: [9x4 double]
    pieces: 9
     order: 4
       dim: 1
ans =
   -1.86769571596165e-005
    1.0498011444416e-007
   -8.33938341141523e-009
   -2.91051424869693e-009
    1.07676430364134e-008
   -6.38050747119223e-009
    4.17366920830394e-009
   -2.98394920120397e-009
    9.06716937295945e-009
ans =
      0.00548859512409308
   -0.000114492023791968
    1.14841135412007e-005
   -1.02496157569476e-006
   -5.39073294817172e-006
    1.07607316057283e-005
   -8.38079080745047e-006
    4.14021681717713e-006
   -4.81163078643476e-006
ans =
          519.826808036436
          520.364218346466
          520.323015182366
          520.328244758348
          520.325036911085
          520.327721910415
          520.330101851213
          520.325861277223
          520.325189863253
ans =
         -155126.664664063
         -103107.774866338
          105026.312475766
          365189.648672407
          625353.150996907
          885515.667724592
          1405847.76985914
          1926173.66458876
          2446500.68213359
```



Then we examine the approximation error introduced by this spline fit.

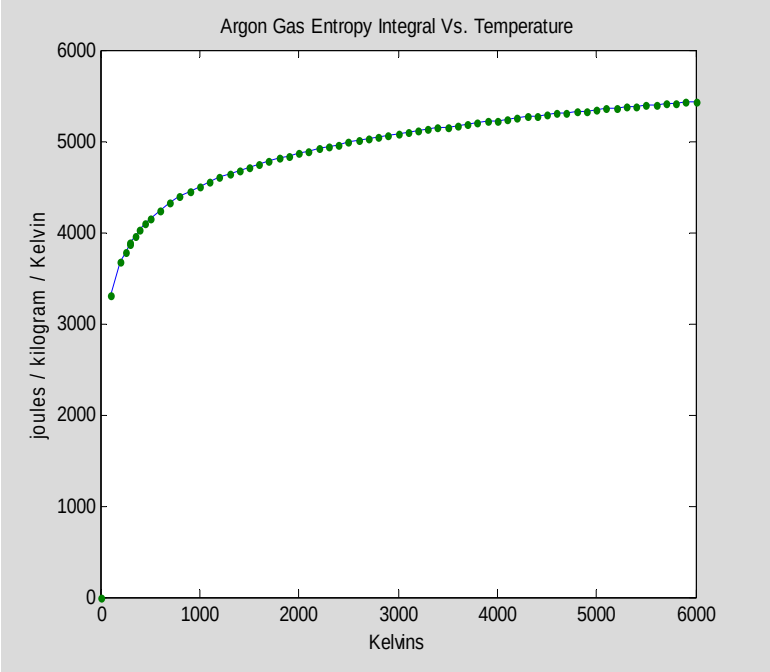
```
H0_Ar_fit=fnval(H0_Ar_pp,linspace(0,7000,500)); plot(TK_H0_Ar,(H0_Ar-
0.039948*fnval(H0_Ar_pp,TK_H0_Ar)/1000) /(max(H0_Ar)-min(H0_Ar)),'.'),
xlabel('Kelvins'),
ylabel('dimensionless'), title('    Normalized Fitting Error in Argon
Enthalpy'), legend('(JANAF data - Spline Fit)/(JANAF data
range)','Location','North')
```



C.3.3 Argon Gas Entropy Integral

Argon entropy integral values are taken from the published JANAF tables as follows. Entropy integral values here are given in joules per mole per Kelvin and are tabulated vs temperature in Kelvins. The mass of a mole of argon is 0.039948 kilogram, a fact used to convert enthalpy to the joules per kilogram per Kelvin form plotted in the graph below.

```
TK_S0_Ar=[0;100;200;250;298.15;300;350;400;450;500;600;700;800;900;1000
;1100;1200;1300;1400;1500;1600;1700;1800;1900;2000;2100;2200;2300;2400;
2500;2600;2700;2800;2900;3000;3100;3200;3300;3400;3500;3600;3700;3800;
3900;4000;4100;4200;4300;4400;4500;4600;4700;4800;4900;5000;5100;5200;
5300;5400;5500;5600;5700;5800;5900;6000];
S0_Ar
=[0;132.137;146.545;151.183;154.845;154.973;158.177;160.953;163.401;
165.591;169.381;172.585;175.361;177.809;179.999;181.98;183.789;185.453;
186.993;188.427;189.769;191.029;192.217;193.341;194.407;195.421;196.388
;
197.312;198.197;199.045;199.86;200.645;201.401;202.13;202.835;203.516;
204.176;204.816;205.436;206.039;206.625;207.194;207.748;208.288;208.815
;
209.328;209.829;210.318;210.796;211.263;211.72;212.167;212.604;213.033;
213.453;213.864;214.268;214.664;215.053;215.434;215.808;216.176;216.538
;216.893; 217.243];
plot(TK_S0_Ar(2:end),S0_Ar(2:end)/0.039948,TK_S0_Ar,S0_Ar/0.039948,'.')
, xlabel('Kelvins'),
ylabel('joules / kilogram / Kelvin'),
title('Argon Gas Entropy Integral Vs. Temperature')
```

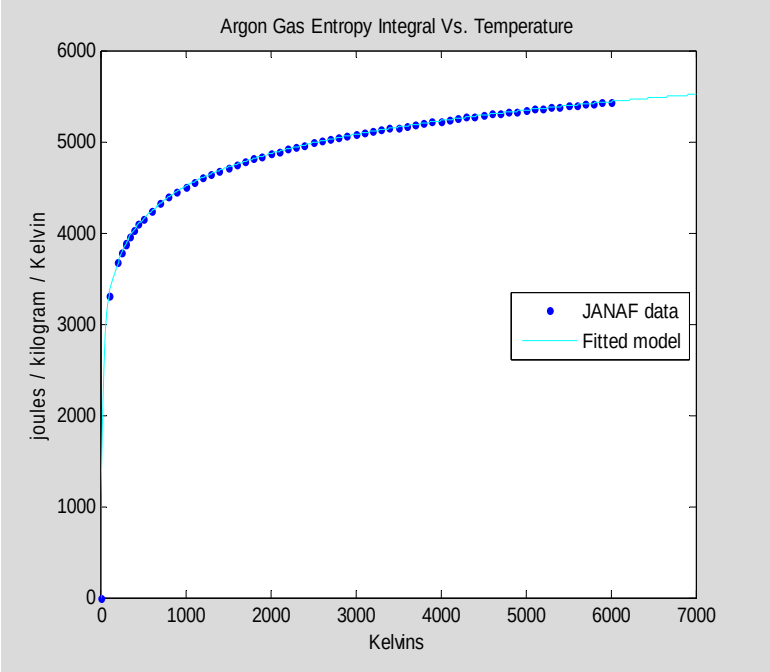


Next, we develop a leasst-squares spline fit to this data using a fixed knot sequence.

```
knots=augknt([0 100 500 1000 1500 2000 3000 4000 5000 6000],4);
S0_Ar_pp=fn2fm(spap2(knots,4,TK_S0_Ar,S0_Ar/0.039948),'pp');
S0_Ar_fit=fnval(S0_Ar_pp,linspace(0,7000,500));
plot(TK_S0_Ar,S0_Ar/0.039948,'.',linspace(0,7000,500), S0_Ar_fit,'-c'),
xlabel('Kelvins'), ylabel('joules / kilogram / Kelvin'), title('Argon
Gas Entropy Integral Vs. Temperature'), legend('JANAF data','Fitted
model','Location','East'),
format long g,
display(S0_Ar_pp),
S0_Ar_pp.coefs(:,1),
S0_Ar_pp.coefs(:,2),
S0_Ar_pp.coefs(:,3),
S0_Ar_pp.coefs(:,4)
```

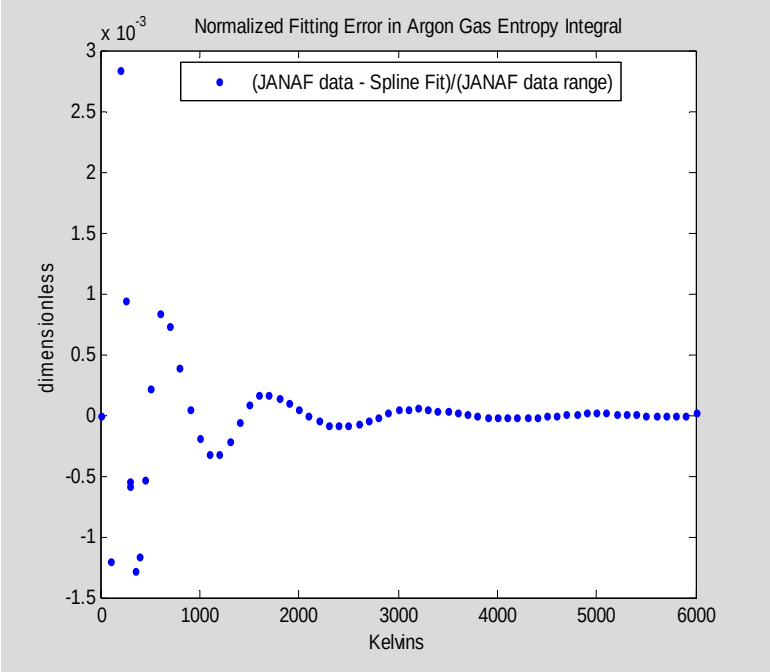
```
S0_Ar_pp =
      form: 'pp'
    breaks: [0 100 500 1000 1500 2000 3000 4000 5000 6000]
     coefs: [9x4 double]
    pieces: 9
     order: 4
       dim: 1
ans =
       0.00283938937589918
     5.40181040798507e-006
     2.03529184454081e-007
     1.33082029182343e-007
```

```
     1.60929688703906e-008
     1.35082404124139e-008
     2.67713523087757e-009
     2.59288338559281e-009
     1.12645363114093e-010
ans =
        -0.858893627649924
      -0.00707681488016913
     -0.000594642390587056
     -0.000289348613905951
     -8.9725570132443e-005
    -6.55861168268561e-005
    -2.50613955896151e-005
    -1.70299898969802e-005
      -9.251339740199e-006
ans =
          90.6378774936673
          4.04083324065806
         0.972250332355587
         0.530254830109099
         0.340717738089908
         0.263061894610258
         0.172414382193788
         0.130322996707188
         0.104041667070005
ans =
                         0
          3314.24084876668
          4143.99963031389
          4506.90534690167
          4716.33086212753
          4866.26995974817
          5077.25397794398
          5227.28409977904
          5343.16998997484
```



Then we examine the residual discrepancies introduced by the spline fit.

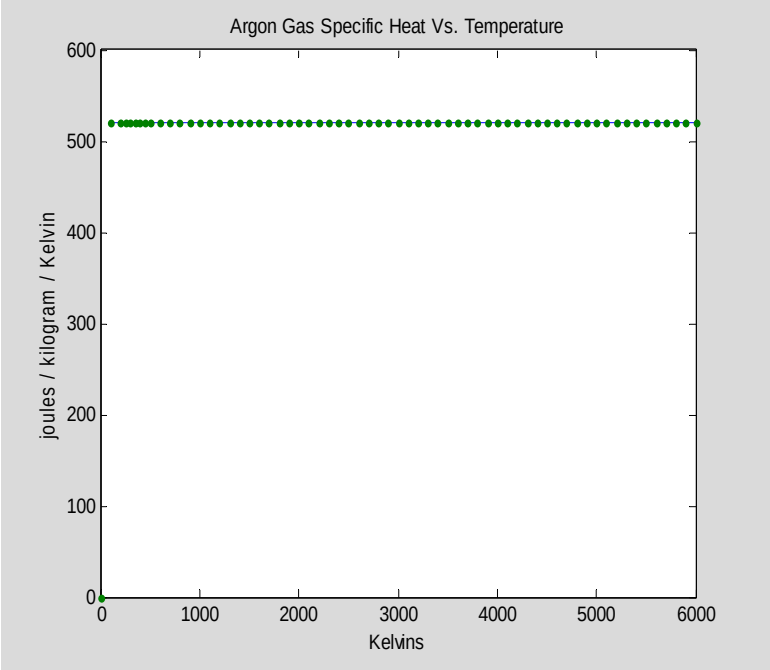
```
S0_Ar_fit=fnval(S0_Ar_pp,linspace(0,7000,500)); plot(TK_S0_Ar,(S0_Ar-
0.039948*fnval(S0_Ar_pp,TK_S0_Ar)) /(max(S0_Ar)-min(S0_Ar)),'.'),
xlabel('Kelvins'),
ylabel('dimensionless'), title('    Normalized Fitting Error in Argon
Gas Entropy Integral'), legend('(JANAF data - Spline Fit)/(JANAF data
range)','Location','North')
```


Normalized Fitting Error in Argon Gas Entropy Integral
x 10^-3
(JANAF data - Spline Fit)/(JANAF data range)
dimensionless
Kelvins
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
0
1000
2000
3000
4000
5000
6000

C.3.4 Argon Gas Specific Heat

Argon gas specific heat at constant pressure is taken from the published JANAF tables as follows. Specific heat values here are given in joules per mole per Kelvin and are tabulated vs temperature in Kelvins. The mass of a mole of argon is 0.039948 kilogram, a fact used to convert specific heat to the joules per kilogram per Kelvin form plotted in the graph below.

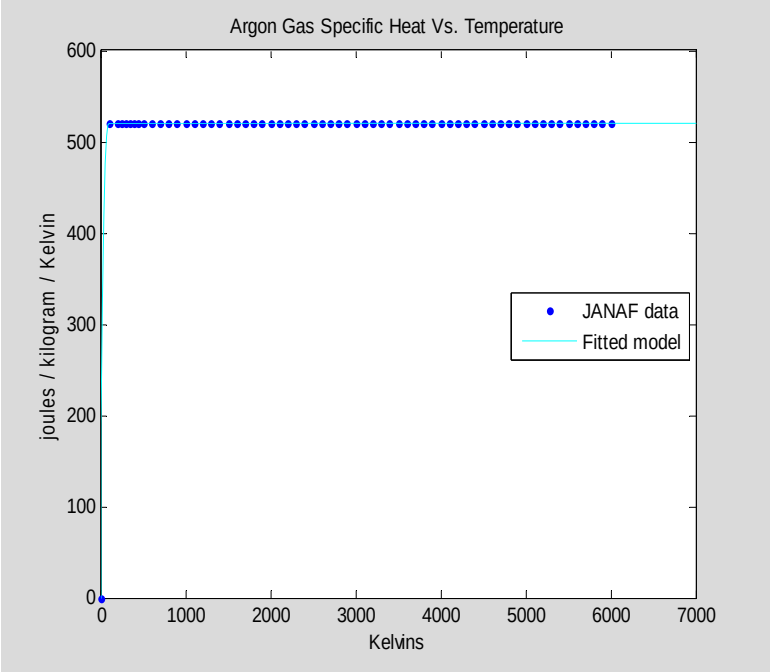
```
TK_Cp_Ar=[0;100;200;250;298.15;300;350;400;450;500;600;700;800;900;1000
;1100;1200;1300;1400;1500;1600;1700;1800;1900;2000;2100;2200;2300;2400;
2500;2600;2700;2800;2900;3000;3100;3200;3300;3400;3500;3600;3700;3800;
3900;4000;4100;4200;4300;4400;4500;4600;4700;4800;4900;5000;5100;5200;
5300;5400;5500;5600;5700;5800;5900;6000];
Cp_Ar=[0;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786
;
20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;
20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;
20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;
20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;
20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;
20.786;20.786;20.786;20.786;20.786];
plot(TK_Cp_Ar(2:end),Cp_Ar(2:end)/0.039948,TK_Cp_Ar,Cp_Ar/0.039948,'.')
,xlabel('Kelvins'),
ylabel('joules / kilogram / Kelvin'),
title('Argon Gas Specific Heat Vs. Temperature')
```



Next, we generate a spline approximation model fitted to the data.

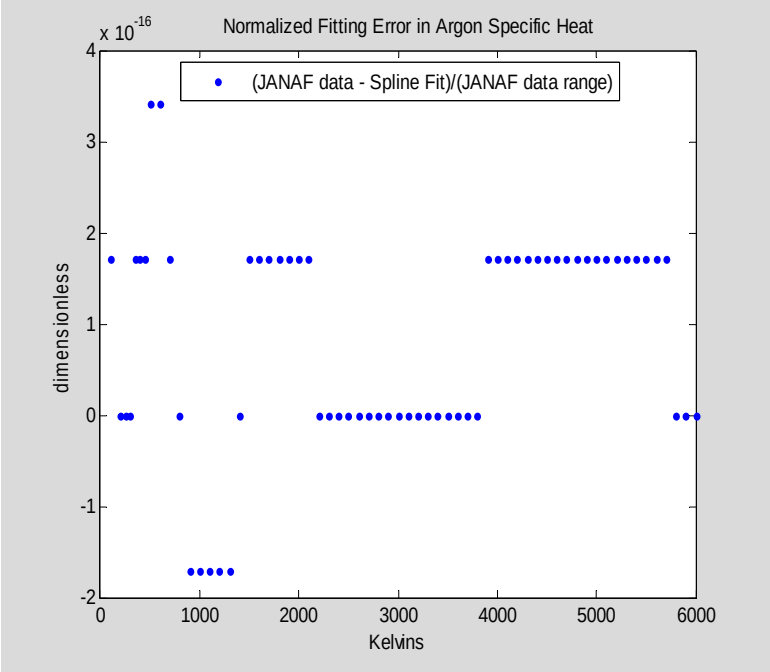
```
knots=augknt([0 100 500 1000 1500 2000 3000 4000 5000 6000],4);
Cp_Ar_pp=fn2fm(spap2(knots,4,TK_Cp_Ar,Cp_Ar/0.039948),'pp');
Cp_Ar_fit=fnval(Cp_Ar_pp,linspace(000,7000,500));
plot(TK_Cp_Ar,Cp_Ar/0.039948,'.',linspace(000,7000,500), Cp_Ar_fit,'-
c'), xlabel('Kelvins'), ylabel('joules / kilogram / Kelvin'),
title('Argon Gas Specific Heat Vs. Temperature'), legend('JANAF
data','Fitted model','Location','East'),
format long g,
display(Cp_Ar_pp),
Cp_Ar_pp.coefs(:,1),
Cp_Ar_pp.coefs(:,2),
Cp_Ar_pp.coefs(:,3),
Cp_Ar_pp.coefs(:,4)

Cp_Ar_pp =
      form: 'pp'
    breaks: [0 100 500 1000 1500 2000 3000 4000 5000 6000]
     coefs: [9x4 double]
    pieces: 9
     order: 4
       dim: 1
ans =
      0.000520326424351657
     1.08335836872065e-020
    -4.01693493283043e-021
     3.06954461848363e-021
    -3.43586887336439e-022
      -9.473903143468e-024
     1.04212934578148e-022
    -1.13686837721616e-022
     1.13686837721616e-022
ans =
        -0.156097927305497
     -9.0002079862946e-018
     3.41060513164848e-018
    -3.41060513164848e-018
     4.54747350886464e-019
    -1.70530256582424e-019
    -1.70530256582424e-019
     1.70530256582424e-019
                        0
ans =
          15.6097927305497
     1.70530256582424e-015
                        0
     6.82121026329696e-016
                        0
     3.41060513164848e-016
                        0
                        0
                        0
ans =
                        0
          520.326424351657
          520.326424351657
```

```
520.326424351657
520.326424351657
520.326424351657
520.326424351657
520.326424351657
520.326424351657
```



Then we examine the approximation error introduced by the spline fit.

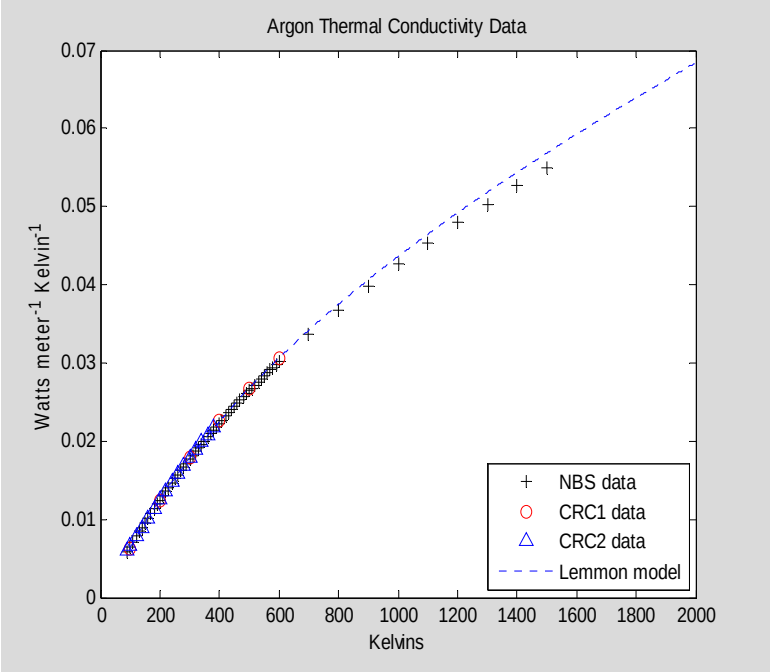
```
Cp_Ar_fit=fnval(Cp_Ar_pp,linspace(0,7000,500)); plot(TK_Cp_Ar,(Cp_Ar-
0.039948*fnval(Cp_Ar_pp,TK_Cp_Ar)) /(max(Cp_Ar)-min(Cp_Ar)),'.'),
xlabel('Kelvins'),
ylabel('dimensionless'), title('    Normalized Fitting Error in Argon
Specific Heat'), legend('(JANAF data - Spline Fit)/(JANAF data
range)','Location','North')
```


Normalized Fitting Error in Argon Specific Heat
x 10^-16
(JANAF data - Spline Fit)/(JANAF data range)
dimensionless
Kelvins
4
3
2
1
0
-1
-2
0
1000
2000
3000
4000
5000
6000

C.3.5 Argon Gas Thermal Conductivity

Thermal conductivity data for argon gas was found from two different parts of the CRC handbook and from the 1955 NBS Circular 564 . In addition, a 2004 paper [Lemmon and Jacobsen] provides comprehensive formulae for argon thermal conductivity. Values here are given in watts per meter per Kelvin, and are tabulated versus temperature in Kelvins. Since the thermal conductivities of gases such as argon are almost independent of pressure in the range of pressures relevant to this work, no pressure dependence is modeled here.

```
TK_ArThrmCondCRC1=[100;200;300;400;500;600];
ArThrmCondCRC1=1e-3* [6.2;12.4;17.9;22.6;26.8;30.6];
TK_ArThrmCondCRC2=[90;100;120;140;160;180;200;220;240;260;280;300;320;3
40;360;380];
ArThrmCondCRC2=1e-3*
[6.0;6.6;7.8;9.0;10.2;11.4;12.5;13.7;14.8;15.8;16.9;17.9;18.9;19.9;20.8
;21.7];
TK_ArThrmCondNBS=[90;100;110;120;130;140;150;160;170;180;190;200;210;22
0;230;240;250;260;270;280;290;300;310;320;330;340;350;360;370;380;390;4
00;410;420;430;440;450;460;470;480;490;500;510;520;530;540;550;560;570;
580;590;600;700;800;900;1000;1100;1200;1300;1400;1500];
ArThrmCondNBS=0.01634*[0.361;0.394;0.438;0.476;0.514;0.550;0.586;0.622;
0.658;0.693;0.728;0.763;0.796;0.830;0.863;0.896;0.927;0.958;0.991;1.022
;1.052;1.081;1.111;1.140;1.169;1.198;1.227;1.255;1.283;1.310;1.337;1.36
3;1.390;1.416;1.442;1.467;1.493;1.518;1.543;1.568;1.592;1.616;1.640;1.6
64;1.688;1.711;1.734;1.757;1.780;1.802;1.824;1.846;2.057;2.254;2.438;2.
611;2.775;2.932;3.081;3.225;3.362];
TK=(100:2000)';
ThrmCondArLemmon=(0.8158* 2.66958E-2*sqrt(39.948*TK)/(0.335)^2
./exp(0.431-0.4623*log(TK/143.2) +0.08406*log(TK/143.2).^2
+0.005341*log(TK/143.2).^3-0.00331*log(TK/143.2).^4 )
-0.4320*(TK/150.687).^0.77)/1000;
plot(TK_ArThrmCondNBS,ArThrmCondNBS,'+k',TK_ArThrmCondCRC1,
ArThrmCondCRC1,'or', TK_ArThrmCondCRC2,ArThrmCondCRC2,
'^b',TK,ThrmCondArLemmon,':b'),
xlabel('Kelvins'),
ylabel('Watts meter^-^1 Kelvin^-^1'),
title('Argon Thermal Conductivity Data'),
legend('NBS data','CRC1 data','CRC2 data','Lemmon
model','Location','SouthEast')
```

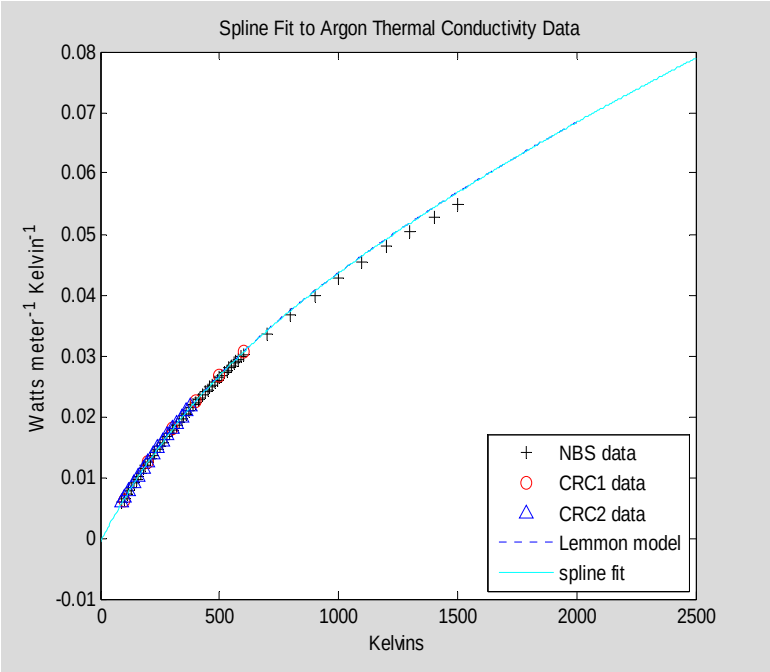


Next we approximate this data, including the extrapolated model points, by using least squares techniques to fit cubic splines with a priori specified knot sequences.

```
knots=augknt([0 500 1000 1500 2000],4);
kth_Ar_pp=fn2fm(spap2(knots,4,[TK_ArThrmCondNBS;  TK_ArThrmCondCRC1;
TK_ArThrmCondCRC2; TK], [ArThrmCondNBS; ArThrmCondCRC1; ArThrmCondCRC2;
ThrmCondArLemmon]),'pp');
kth_Ar_fit=fnval(kth_Ar_pp,linspace(000,2500,500));
plot(TK_ArThrmCondNBS,ArThrmCondNBS,'+k',TK_ArThrmCondCRC1,
ArThrmCondCRC1,'or', TK_ArThrmCondCRC2,ArThrmCondCRC2,
'^b',TK,ThrmCondArLemmon,':b',linspace(000,2500,500),kth_Ar_fit,'-c'),
xlabel('Kelvins'),
ylabel('Watts meter^-^1 Kelvin^-^1'),
title('  Spline Fit to Argon Thermal Conductivity Data '),
legend('NBS data','CRC1 data','CRC2 data','Lemmon model','spline
fit','Location','SouthEast'),
format long g,
display(kth_Ar_pp),
kth_Ar_pp.coefs(:,1),
kth_Ar_pp.coefs(:,2),
kth_Ar_pp.coefs(:,3),
kth_Ar_pp.coefs(:,4)
```

```
kth_Ar_pp =
      form: 'pp'
    breaks: [0 500 1000 1500 2000]
     coefs: [4x4 double]
    pieces: 4
     order: 4
```

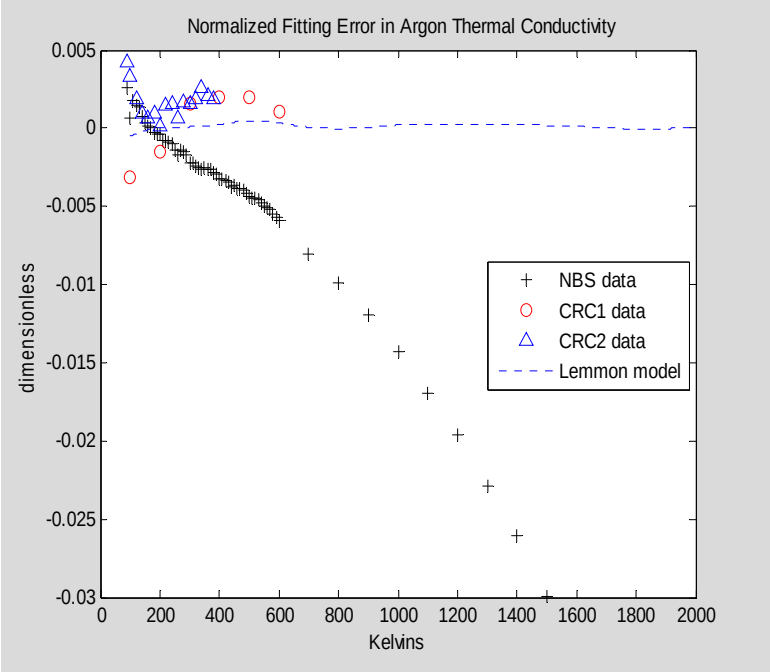
```
       dim: 1
ans =
     2.58959511719021e-011
     6.59376178597951e-012
     2.10850059204909e-012
     6.33292311079228e-013
ans =
    -5.48845783903993e-008
    -1.60406516325461e-008
    -6.15000895357691e-009
    -2.98725806550341e-009
ans =
     7.55998625450829e-005
     4.01372475336103e-005
     2.90419172405488e-005
     2.44732837310088e-005
ans =
    -0.000640792446023078
       0.0266749881254063
       0.0435576692073224
       0.0568046881632087
```



Then we examine the approximation error introduced by the spline fit.

```
ArThrmCondRange=max(ThrmCondArLemmon)-min(ArThrmCondNBS);
plot(TK_ArThrmCondNBS,(ArThrmCondNBS-
fnval(kth_Ar_pp,TK_ArThrmCondNBS))/ArThrmCondRange,'+k',
TK_ArThrmCondCRC1, (ArThrmCondCRC1-
fnval(kth_Ar_pp,TK_ArThrmCondCRC1))/ArThrmCondRange,'or',
TK_ArThrmCondCRC2,(ArThrmCondCRC2-
fnval(kth_Ar_pp,TK_ArThrmCondCRC2))/ArThrmCondRange,'^b',
TK,(ThrmCondArLemmon-fnval(kth_Ar_pp,TK))/ArThrmCondRange,':b'),
```

```
xlabel('Kelvins'),
ylabel('dimensionless'),
title('  Normalized Fitting Error in Argon Thermal Conductivity'),
legend('NBS data','CRC1 data','CRC2 data','Lemmon
model','Location','East')
```

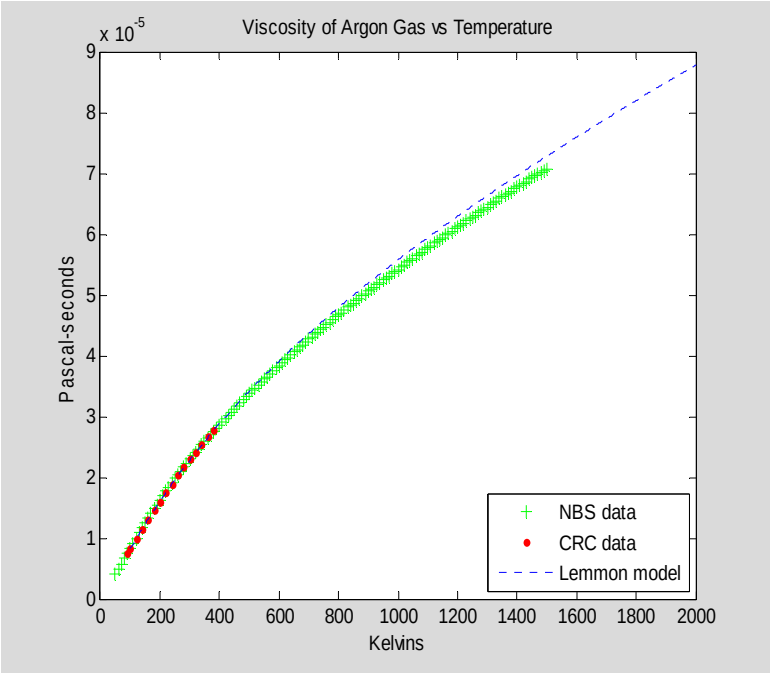


C.3.6 Argon Gas Viscosity

Viscosity data for argon gas was found in the CRC handbook, in NBS Circular 564 (which dates from 1955) and in a 2004 paper by Lemmon et al. Values here are given in Pascal-seconds, and are tabulated versus temperature in Kelvins. Since the viscosities of gases such as argon are almost independent of pressure in the range of pressures relevant to this work, no pressure dependence is modeled here.

```
TK_ArViscNBS=[50;60;70;80;90;100;110;120;130;140;150;160;170;180;190;20
0;210;220;230;240;250;260;270;280;290;300;310;320;330;340;350;360;370;3
80;390;400;410;420;430;440;450;460;470;480;490;500;510;520;530;540;550;
560;570;580;590;600;610;620;630;640;650;660;670;680;690;700;710;720;730
;740;750;760;770;780;790;800;810;820;830;840;850;860;870;880;890;900;91
0;920;930;940;950;960;970;980;990;1000;1010;1020;1030;1040;1050;1060;10
70;1080;1090;1100;1110;1120;1130;1140;1150;1160;1170;1180;1190;1200;121
0;1220;1230;1240;1250;1260;1270;1280;1290;1300;1310;1320;1330;1340;1350
;1360;1370;1380;1390;1400;1410;1420;1430;1440;1450;1460;1470;1480;1490;
1500];
ArViscNBS=2.125E-5*
[0.1965;0.2336;0.2718;0.3108;0.3506;0.3905;0.4302;0.4696;0.5082;0.5468;
0.5849;
0.6221;0.6588;0.6946;0.7299;0.7647;0.7991;0.8315;0.8649;0.8974;0.9289;
0.9600;0.9906;1.0207;1.0504;1.0795;1.1082;1.1365;1.16;1.1920;1.219;1.24
5;
1.272;1.298;1.323;1.349;1.374;1.398;1.423;1.448;1.472;1.496;1.519;1.542
;1.565;
1.588;1.611;1.633;1.655;1.677;1.699;1.721;1.742;1.764;1.785;1.806;1.827
;1.847;
1.868;1.888;1.908;1.928;1.948;1.968;1.988;2.008;2.028;2.047;2.067;2.086
;2.105;
2.123;2.142;2.161;2.180;2.198;2.216;2.235;2.254;2.272;2.290;2.308;2.326
;2.344;
2.361;2.379;2.396;2.414;2.431;2.449;2.466;2.483;2.500;2.517;2.534;2.551
;2.568;
2.585;2.602;2.618;2.634;2.651;2.668;2.684;2.701;2.717;2.733;2.750;2.766
;2.782;
2.798;2.814;2.830;2.845;2.861;2.876;2.892;2.908;2.924;2.940;2.956;2.971
;2.987;3.002;3.018;3.034;3.049;3.065;3.081;3.097;3.112;3.127;3.142;3.15
7;3.172;3.186;
3.201;3.216;3.230;3.245;3.259;3.274;3.288;3.303;3.317;3.331];
TK_ArViscCRC=[90;100;120;140;160;180;200;220;240;260;280;300;320;340;36
0;380];
ArViscCRC=1e-6*
[7.5;8.2;9.8;11.4;13.0;14.5;16.0;17.5;18.9;20.3;21.6;22.9;24.2;25.4;26.
6;27.8];
TK=(100:2000)';
ViscArLemmon=2.66958E-8*sqrt(39.948*TK)/(0.335)^2./exp(0.431-
0.4623*log(TK/143.2)+0.08406*log(TK/143.2).^2+0.005341*log(TK/143.2).^3
-0.00331*log(TK/143.2).^4);
plot(TK_ArViscNBS,ArViscNBS,'+g',TK_ArViscCRC,ArViscCRC,'.r',
TK,ViscArLemmon,':b'),
title('Viscosity of Argon Gas vs Temperature'),
```

```
ylabel('Pascal-seconds'),
xlabel('Kelvins'),
legend('NBS data','CRC data','Lemmon model','Location','SouthEast')
```

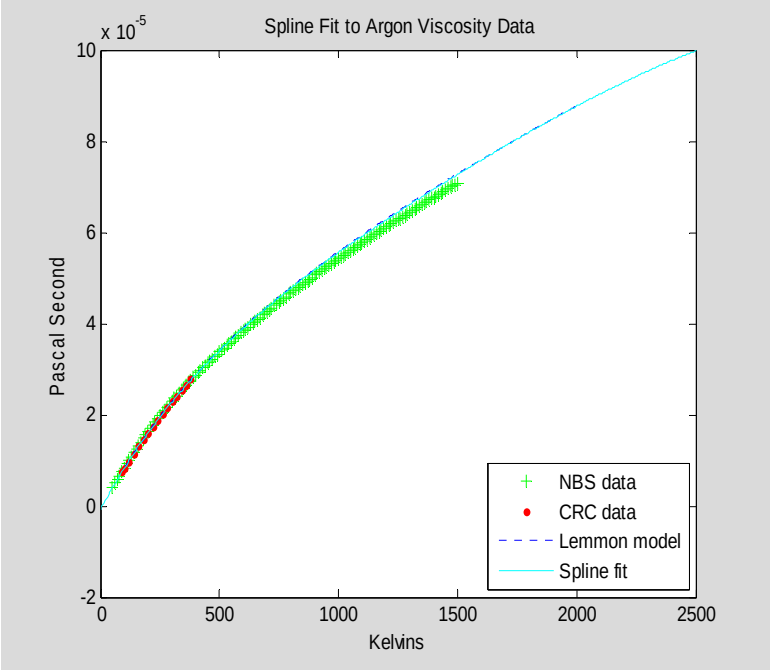


Next we approximate this data by using least squares techniques to fit a cubic spline with knot sequences specified a priori.

```
knots=augknt([0 500 1000 1500 2000],4);
mu_Ar_pp=fn2fm(spap2(knots,4,[TK_ArViscNBS; TK_ArViscCRC; TK],
[ArViscNBS; ArViscCRC; ViscArLemmon]),'pp');
mu_Ar_fit=fnval(mu_Ar_pp,linspace(000,2500,500));
plot(TK_ArViscNBS,ArViscNBS,'+g',TK_ArViscCRC,ArViscCRC,'.r',TK,ViscArL
emmon,':b',linspace(000,2500,500),mu_Ar_fit,'-c')
xlabel('Kelvins'),
ylabel('Pascal Second'),
title('  Spline Fit to Argon Viscosity Data '),
legend('NBS data','CRC data','Lemmon model','Spline
fit','Location','SouthEast')
format long g,
display(mu_Ar_pp),
mu_Ar_pp.coefs(:,1),
mu_Ar_pp.coefs(:,2),
mu_Ar_pp.coefs(:,3),
mu_Ar_pp.coefs(:,4)
```

```
mu_Ar_pp =
      form: 'pp'
    breaks: [0 500 1000 1500 2000]
     coefs: [4x4 double]
    pieces: 4
```

```
     order: 4
       dim: 1
ans =
     3.40044524999721e-014
     8.58057752977249e-015
     2.81766298833804e-015
    -1.71211390699328e-015
ans =
     -7.1216312951265e-011
    -2.02096342013069e-011
    -7.33876790664809e-012
    -3.11227342414123e-012
ans =
     9.66890140720963e-008
     5.09760404958105e-008
     3.72018394418329e-008
     3.19763187764384e-008
ans =
    -7.64588468456918e-007
     3.40263968922715e-005
     5.55345807810716e-005
     7.26530163988682e-005
```



C.4 Carbon Dioxide Gas Properties

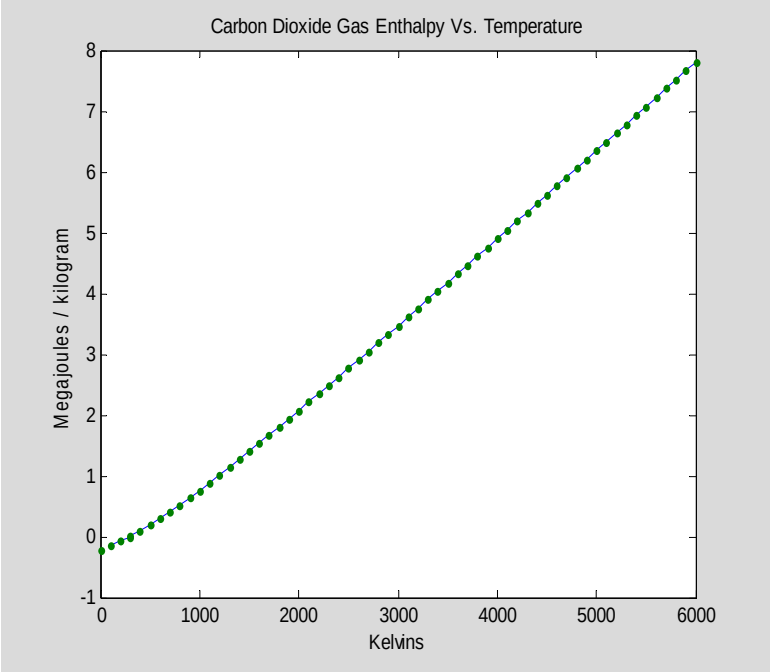
C.4.1 Carbon Dioxide Gas Density

$$\rho = \frac{p}{RT} \text{ where } R = \frac{\mathcal{R}}{\hat{M}} = \frac{8.31361 \text{ Joule/mole/Kelvin}}{0.0440098 \text{ kg/mole}} = 188.9036 \text{ Joule kg}^{-1} \text{ Kelvin}^{-1}$$

C.4.2 Carbon Dioxide Gas Enthalpy

Carbon Dioxide enthalpy is taken from the published JANAF tables as follows. Enthalpy values here are given in kilojoules per mole and are tabulated vs temperatures in Kelvins. The mass of a mole of carbon dioxide is 0.0440098 kilogram, a fact used to convert enthalpy to the megajoules per kilogram form plotted in the graph below.

```
TK_H0_CO2=[0;100;200;298.15;300;400;500;600;700;800;900;1000;1100;1200;
1300;1400;1500;1600;1700;1800;1900;2000;2100;2200;2300;2400;2500;2600;
2700;2800;2900;3000;3100;3200;3300;3400;3500;3600;3700;3800;3900;4000;
4100;4200;4300;4400;4500;4600;4700;4800;4900;5000;5100;5200;5300;5400;
5500;5600;5700;5800;5900;6000];
H0_CO2 =[-9.364;-6.456;-3.414;0;0.069;4.003;8.305;12.907;17.754;22.806;
28.03;33.397;38.884;44.473;50.148;55.896;61.705;67.569;73.48;79.431;
85.419;91.439;97.488;103.562;109.66;115.779;121.917;128.073;134.246;
140.433;146.636;152.852;159.081;165.321;171.573;177.836;184.109;190.393
;196.686;202.989;209.301;215.622;221.951;228.29;234.637;240.991;247.354
;
253.725;260.103;266.489;272.882;279.283;285.691;292.109;298.535;304.971
;311.416;317.87;324.334; 330.806;337.288;343.779];
plot(TK_H0_CO2(2:end),H0_CO2(2:end)/0.0440098/1000,TK_H0_CO2,H0_CO2/0.0
440098/1000,'.'),
xlabel('Kelvins'),
ylabel('Megajoules / kilogram'),
title('Carbon Dioxide Gas Enthalpy Vs. Temperature')
```

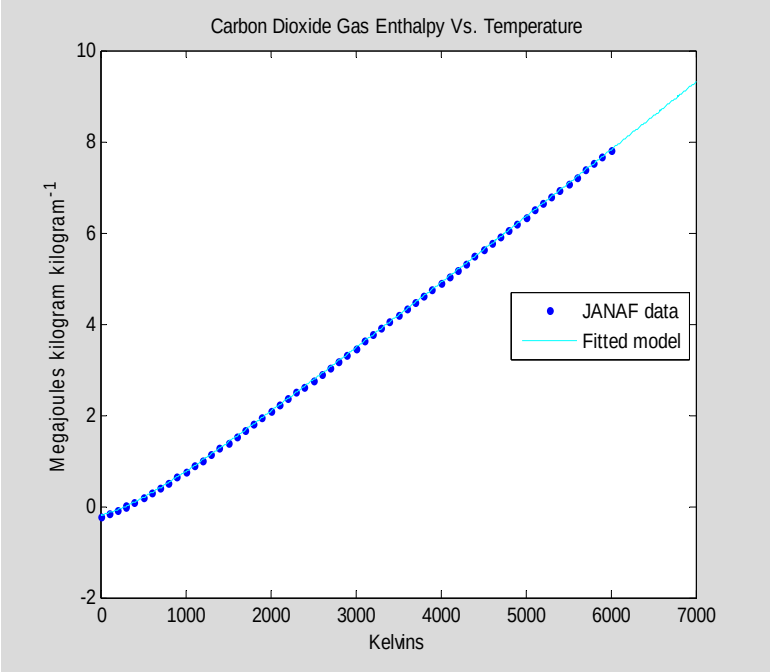


Next we approximate this data by using least squares techniques to fit cubic splines with a priori specified knot sequences.

```
knots=augknt([0 100 500 1000 1500 2000 3000 4000 5000 6000],4);
H0_CO2_pp=fn2fm(spap2(knots,4,TK_H0_CO2,H0_CO2/0.0440098*1000),'pp');
H0_CO2_fit=fnval(H0_CO2_pp,linspace(000,7000,500));
plot(TK_H0_CO2,H0_CO2/0.0440098/1000,'.',linspace(000,7000,500),
H0_CO2_fit/1e6,'-c'), xlabel('Kelvins'), ylabel('Megajoules kilogram
kilogram^-^1'), title('Carbon Dioxide Gas Enthalpy Vs. Temperature'),
legend('JANAF data','Fitted model','Location','East'),
format long g,
display(H0_CO2_pp),
H0_CO2_pp.coefs(:,1),
H0_CO2_pp.coefs(:,2),
H0_CO2_pp.coefs(:,3),
H0_CO2_pp.coefs(:,4)
```

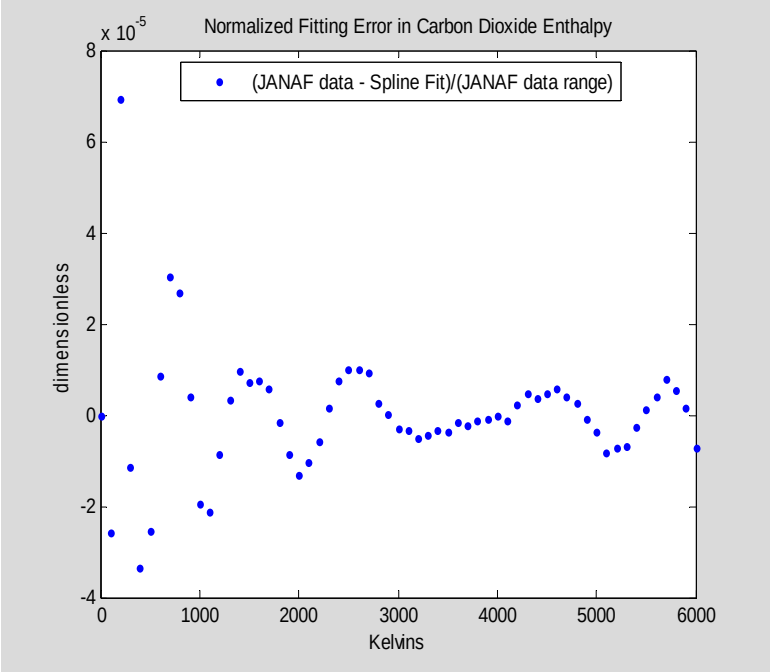
```
H0_CO2_pp =
      form: 'pp'
    breaks: [0 100 500 1000 1500 2000 3000 4000 5000 6000]
     coefs: [9x4 double]
    pieces: 9
     order: 4
       dim: 1
ans =
         0.0107326301178543
      -0.000275726019996787
      -0.000133701237860417
     -4.16462216642564e-005
     -2.07122355094936e-005
```

```
     -5.2765267781502e-006
    -1.24023079762947e-006
    -3.60503394604128e-007
     1.36239384044052e-006
ans =
         -2.56557035988582
         0.654218675470476
         0.323347451474332
         0.122795594683707
        0.0603262621873232
        0.0292579089230821
        0.0134283285886315
       0.00970763619574564
       0.00862612601193041
ans =
          812.060914924536
          620.925746483001
          1011.95219726092
          1235.02372033994
          1326.58464877546
          1371.37673433066
          1414.06297184238
          1437.19893662675
          1455.53269883443
ans =
         -212770.791959973
         -146487.773948523
          188911.04744016
          759011.354206653
          1402016.33533952
          2077801.19583539
          3473159.31231099
          4899410.38194436
          6345956.45137225
```



Then we examine the approximation error introduced by this spline fit.

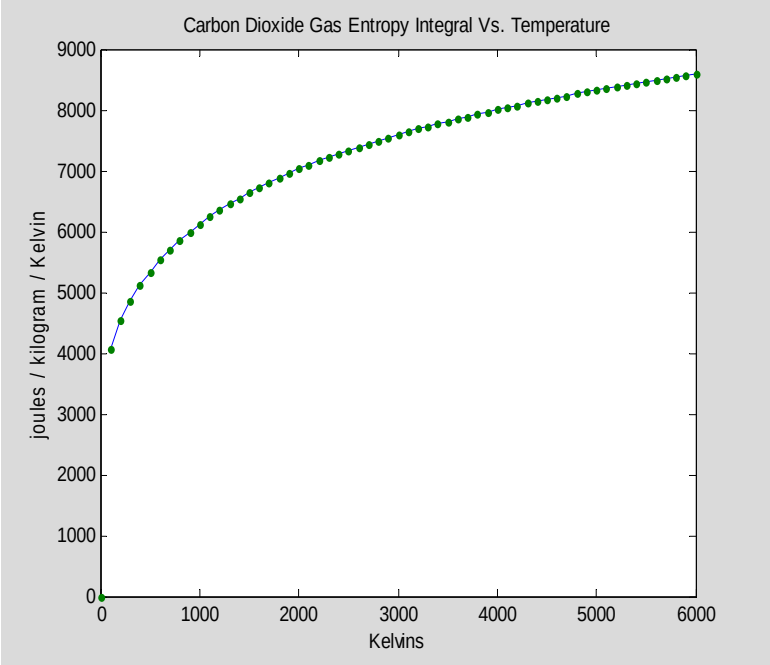
```
H0_CO2_fit=fnval(H0_CO2_pp,linspace(0,7000,500));
plot(TK_H0_CO2,(H0_CO2-0.0440098*fnval(H0_CO2_pp,TK_H0_CO2)/1000)
/(max(H0_CO2)-min(H0_CO2)),'.'),
xlabel('Kelvins'),
ylabel('dimensionless'), title('    Normalized Fitting Error in Carbon
Dioxide Enthalpy'), legend('(JANAF data - Spline Fit)/(JANAF data
range)','Location','North')
```


Normalized Fitting Error in Carbon Dioxide Enthalpy
x 10^-5
(JANAF data - Spline Fit)/(JANAF data range)
dimensionless
Kelvins
8
6
4
2
0
-2
-4
0
1000
2000
3000
4000
5000
6000

C.4.3 Carbon Dioxide Gas Entropy Integral

Carbon Dioxide entropy integral values are taken from the published JANAF tables as follows. Entropy integral values here are given in joules per mole per Kelvin and are tabulated vs temperature in Kelvins. The mass of a mole of argon is 0.0440098 kilogram, a fact used to convert enthalpy to the joules per kilogram per Kelvin form plotted in the graph below.

```
TK_S0_CO2=[0;100;200;298.15;300;400;500;600;700;800;900;1000;1100;1200;
1300;1400;1500;1600;1700;1800;1900;2000;2100;2200;2300;2400;2500;2600;
2700;2800;2900;3000;3100;3200;3300;3400;3500;3600;3700;3800;3900;4000;
4100;4200;4300;4400;4500;4600;4700;4800;4900;5000;5100;5200;5300;5400;
5500;5600;5700;5800;5900;6000];
S0_CO2=[0;179.009;199.975;213.795;214.025;225.314;234.901;243.283;
250.75;257.494;263.645;269.299;274.528;279.39;283.932;288.191;292.199;
295.983;299.566;302.968;306.205;309.293;312.244;315.07;317.781;320.385;
322.89;325.305;327.634;329.885;332.061;334.169;336.211;338.192;340.116;
341.986;343.804;345.574;347.299;348.979;350.619;352.219;353.782;355.31;
356.803;358.264;359.694;361.094;362.466;363.81;365.128;366.422;367.691;
368.937; 370.161;371.364;372.547;373.709;374.853;375.979;377.087;
378.178];
plot(TK_S0_CO2(2:end),S0_CO2(2:end)/0.0440098,TK_S0_CO2,S0_CO2/0.044009
8,'.'), xlabel('Kelvins')
ylabel('joules / kilogram / Kelvin'),
title('Carbon Dioxide Gas Entropy Integral Vs. Temperature')
```

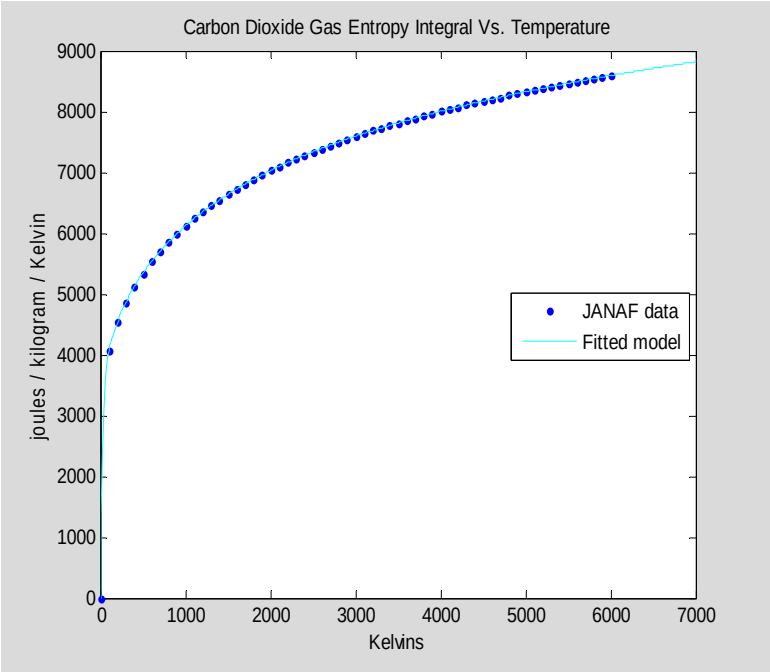


Next, we develop a least-squares spline fit to this data using a fixed knot sequence.

```
knots=augknt([0 100 500 1000 1500 2000 3000 4000 5000 6000],4);
```

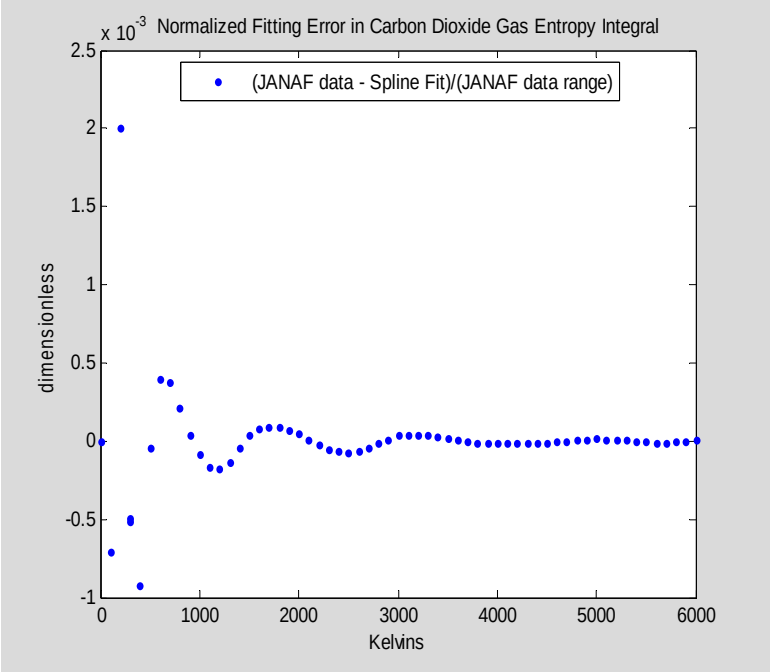
```
S0_CO2_pp=fn2fm(spap2(knots,4,TK_S0_CO2,S0_CO2/0.0440098),'pp');
S0_CO2_fit=fnval(S0_CO2_pp,linspace(0,7000,500));
plot(TK_S0_CO2,S0_CO2/0.0440098,'.',linspace(0,7000,500), S0_CO2_fit,'-
c'), xlabel('Kelvins'), ylabel('joules / kilogram / Kelvin'),
title('Carbon Dioxide Gas Entropy Integral Vs. Temperature'),
legend('JANAF data','Fitted model','Location','East'),
format long g,
display(S0_CO2_pp),
S0_CO2_pp.coefs(:,1),
S0_CO2_pp.coefs(:,2),
S0_CO2_pp.coefs(:,3),
S0_CO2_pp.coefs(:,4)


S0_CO2_pp =
      form: 'pp'
    breaks: [0 100 500 1000 1500 2000 3000 4000 5000 6000]
     coefs: [9x4 double]
    pieces: 9
     order: 4
       dim: 1
ans =
       0.00348237135099007
     5.19469384086673e-006
     2.88512295280341e-007
     1.78234000197562e-007
      5.3480428858403e-008
     2.75090625172642e-008
     8.39820025706481e-009
      5.6144152469153e-009
      1.9032303889362e-009
ans =
         -1.05187574122065
      -0.00716433592362781
     -0.000930703314587747
     -0.000497934871667227
     -0.000230583871370884
     -0.000150363228083283
    -6.78360405314897e-005
    -4.26414397602943e-005
    -2.57981940195477e-005
ans =
          111.099260932629
          5.19525321820124
          1.95723752291503
          1.24291842978754
         0.878659058268482
         0.688185508541401
         0.469986239926629
         0.359508759634843
            0.291069125855
ans =
                         0
          4073.54003204647
```

```
5337.80797736198
6119.81494708261
6639.06969408426
7027.43830898309
7592.76965195847
8003.31805161067
8325.79978673214
```



Then we examine the residual discrepancies introduced by the spline fit.

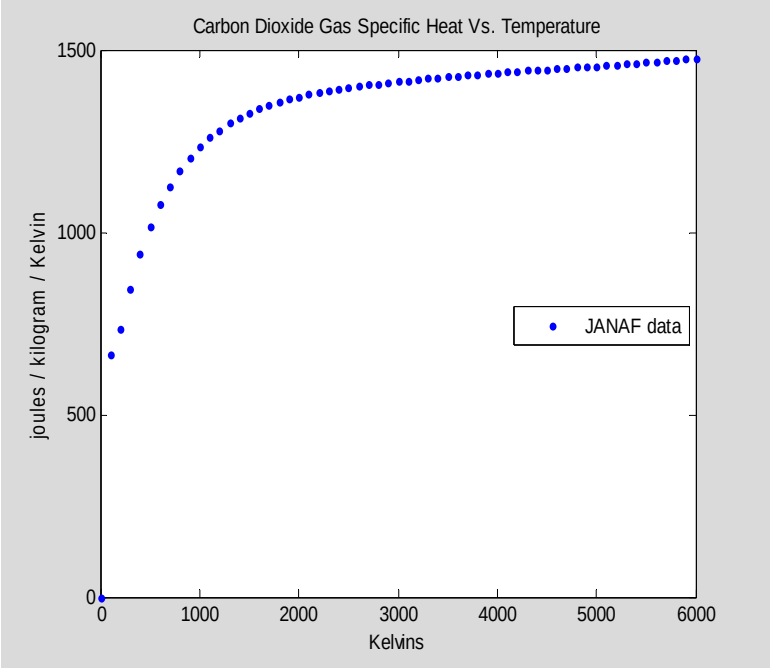
```
S0_CO2_fit=fnval(S0_CO2_pp,linspace(0,7000,500));
plot(TK_S0_CO2,(S0_CO2-0.0440098*fnval(S0_CO2_pp,TK_S0_CO2))
/(max(S0_CO2)-min(S0_CO2)),'.'),
xlabel('Kelvins'),
ylabel('dimensionless'), title('    Normalized Fitting Error in Carbon
Dioxide Gas Entropy Integral'), legend('(JANAF data - Spline
Fit)/(JANAF data range)','Location','North'),
```


x 10^-3 Normalized Fitting Error in Carbon Dioxide Gas Entropy Integral
(JANAF data - Spline Fit)/(JANAF data range)
dimensionless
Kelvins
2.5
2
1.5
1
0.5
0
-0.5
-1
0
1000
2000
3000
4000
5000
6000

C.4.4 Carbon Dioxide Specific Heat

Carbon Dioxide gas specific heat at constant pressure (Cp) is taken from the published JANAF tables as follows. Specific heat values here are given in joules per mole per Kelvin and are tabulated vs temperature in Kelvins. The mass of a mole of carbon dioxide is 0.0440098 kilogram, a fact used to convert specific heat to the joules per kilogram per Kelvin form plotted in the graph below.

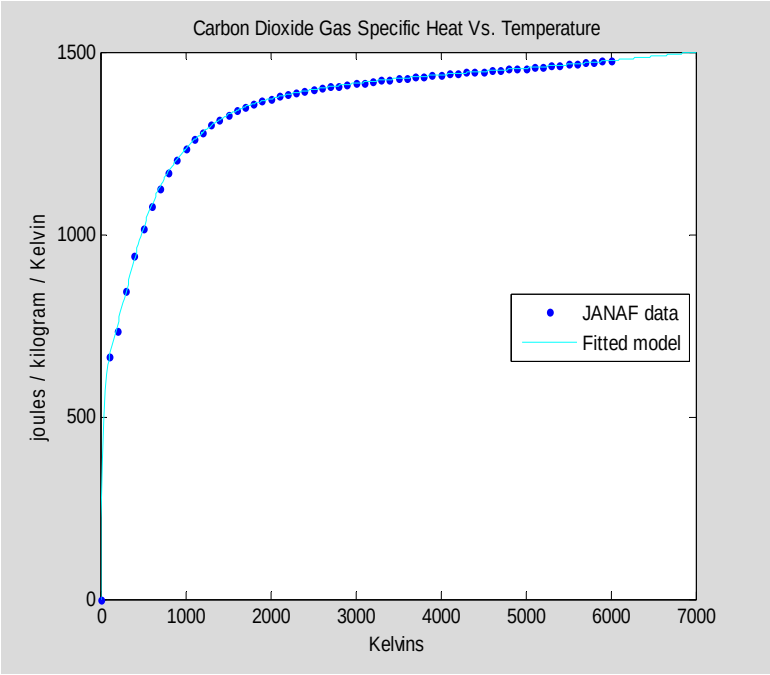
```
TK_Cp_CO2=[0;100;200;298.15;300;400;500;600;700;800;900;1000;1100;1200;
1300;1400;1500;1600;1700;1800;1900;2000;2100;2200;2300;2400;2500;2600;2
700;2800;2900;3000;3100;3200;3300;3400;3500;3600;3700;3800;3900;4000;41
00;4200;4300;4400;4500;4600;4700;4800;4900;5000;5100;5200;5300;5400;550
0;5600;5700;5800;5900; 6000];
Cp_CO2=[0;29.208;32.359;37.129;37.221;41.325;44.627;47.321;49.564;51.43
4;52.999;54.308;55.409;56.342;57.137;57.802;58.379;58.886;59.317;59.701
;60.049;60.35;60.622;60.865;61.086;61.287;61.471;61.647;61.802;61.952;6
2.095;62.229;62.347;62.462;62.573;62.681;62.785;62.884;62.98;63.074;63.
166;63.254;63.341;63.426;63.509;63.588;63.667;63.745;63.823;63.893;63.9
68;64.046;64.128;64.22;64.312;64.404;64.496;64.588;64.68;64.772;64.865;
64.957];
plot(TK_Cp_CO2,Cp_CO2/0.0440098,'.'),xlabel('Kelvins'),
ylabel('joules / kilogram / Kelvin'),
title('Carbon Dioxide Gas Specific Heat Vs. Temperature'),
legend('JANAF data','Location','East')
```



Next, we generate a spline approximation model fitted to the data.

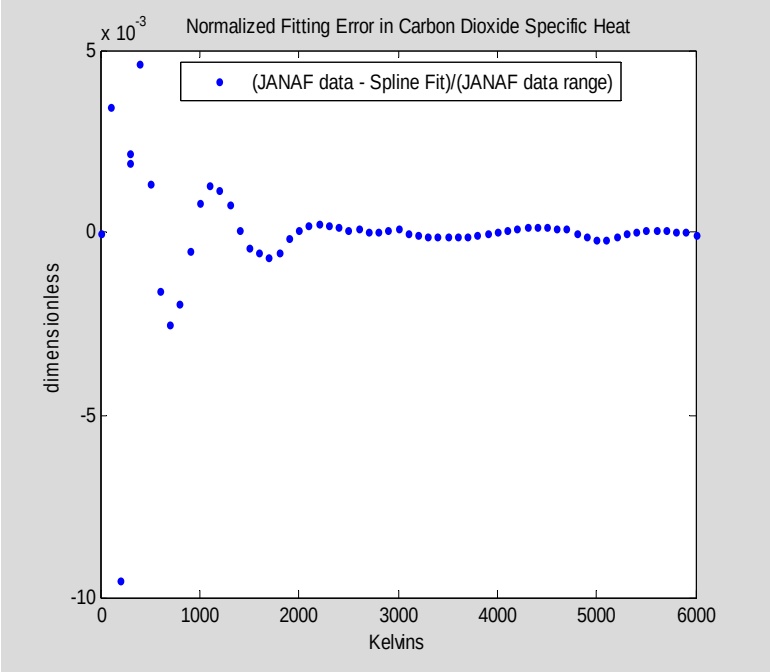
```
knots=augknt([0 100 500 1000 1500 2000 3000 4000 5000 6000],4);
Cp_CO2_pp=fn2fm(spap2(knots,4,TK_Cp_CO2,Cp_CO2/0.0440098),'pp');
Cp_CO2_fit=fnval(Cp_CO2_pp,linspace(000,7000,500));
plot(TK_Cp_CO2,Cp_CO2/0.0440098,'.',linspace(000,7000,500),
Cp_CO2_fit,'-c'), xlabel('Kelvins'), ylabel('joules / kilogram /
Kelvin'), title('Carbon Dioxide Gas Specific Heat Vs. Temperature'),
legend('JANAF data','Fitted model','Location','East'),
format long g,
display(Cp_CO2_pp),
Cp_CO2_pp.coefs(:,1),
Cp_CO2_pp.coefs(:,2),
Cp_CO2_pp.coefs(:,3),
Cp_CO2_pp.coefs(:,4)

Cp_CO2_pp =
      form: 'pp'
    breaks: [0 100 500 1000 1500 2000 3000 4000 5000 6000]
     coefs: [9x4 double]
    pieces: 9
     order: 4
       dim: 1
ans =
      0.000575314506242188
    -1.01092237647369e-006
     4.21180499206655e-007
     3.39193603544049e-008
     5.31776873392503e-008
     6.00664609673335e-009
      1.6691240085583e-009
     1.46697941890937e-009
    -1.78241415538877e-010
ans =
        -0.172169142761837
      0.000425209110819772
      -0.00078789774094866
     -0.000156126992138678
     -0.000105247951607073
    -2.54814205981964e-005
    -7.46148230799611e-006
    -2.45411028232274e-006
     1.94682797440418e-006
ans =
          18.0496670920768
          0.87527372697509
         0.730198274923534
         0.258185908379866
         0.127498436506993
        0.0621337504043581
        0.0291908474981649
        0.0192752549078484
        0.0187679725999319
ans =
                         0
```

```
  658.5897878315
 1012.03370425838
 1232.80596888382
 1327.10709508338
 1371.19153635252
 1413.85051225541
 1437.24900145414
 1455.53712549857
```



Then we examine the approximation error introduced by the spline fit.

```
Cp_CO2_fit=fnval(Cp_CO2_pp,linspace(0,7000,500));
plot(TK_Cp_CO2,(Cp_CO2-0.0440098*fnval(Cp_CO2_pp,TK_Cp_CO2))
/(max(Cp_CO2)-min(Cp_CO2)),'.'),
xlabel('Kelvins'),
ylabel('dimensionless'), title('    Normalized Fitting Error in Carbon
Dioxide Specific Heat'), legend('(JANAF data - Spline Fit)/(JANAF data
range)','Location','North'),
```

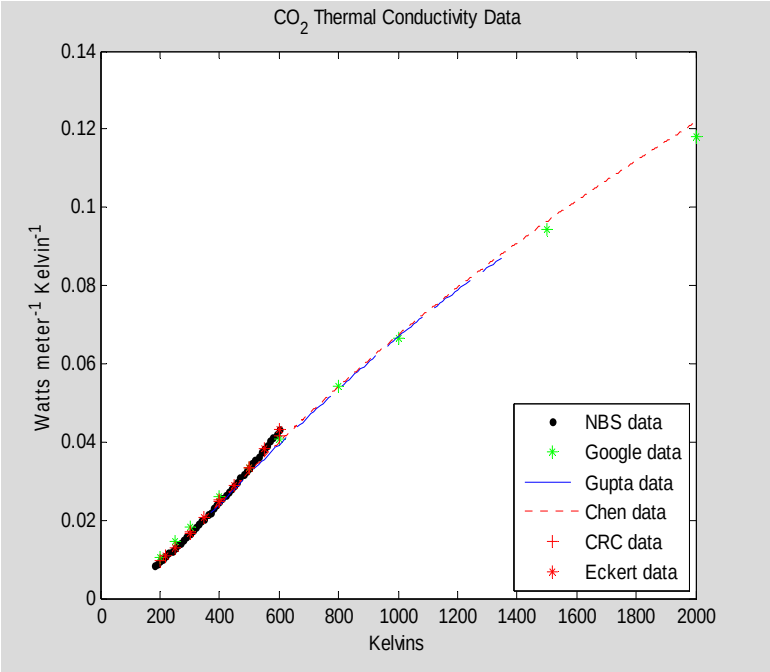

Normalized Fitting Error in Carbon Dioxide Specific Heat
x 10^-3
(JANAF data - Spline Fit)/(JANAF data range)
dimensionless
Kelvins
5
0
-5
-10
0
1000
2000
3000
4000
5000
6000

C.4.5 Carbon Dioxide Gas Thermal Conductivity

Tabulated thermal conductivity data for carbon dioxide gas were found in the CRC Handbook [Lide ed. 1999, 6-184],], in NBS Circular 564, and in the book by Eckert and Drake [Eckert and Drake 1972, 782]. In addition, polynomial representations of CO2 thermal conductivity test data have been given in published refereed papers, including one covering the temperature range from 350 Kelvins to 2000 Kelvins

[Chen, Jain, Saxena 1975] and another covering the temperature range from 100C (=373 Kelvins) to 1075C (=1348 Kelvins) [Gupta and Saxena, 1970].. Finally, a Google search of the internet found CO2 thermal conductivity tabulated at the following internet address: http://www.ipmnet.ru/~yarem/database/co2,html Data from all of these sources have been entered here and are plotted below to simplify comparisons between them. Values here are given in watts per meter per Kelvin, and are plotted versus temperature in Kelvins. Since the thermal conductivities of gases such as CO2 are almost independent of pressure in the range of pressures relevant to this work, no pressure dependence is modeled here.

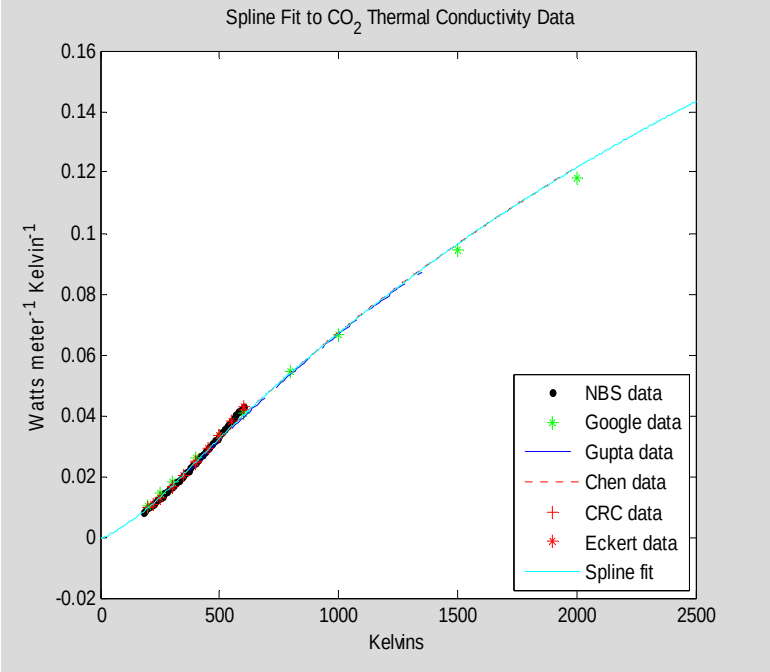
```
TK_CO2ThrmCondChen=linspace(350,2000,500)';
CO2ThrmCondChen=1e-3*(-10.280+0.09427*TK_CO2ThrmCondChen-0.1937E-
4*TK_CO2ThrmCondChen.^2+0.2619E-8*TK_CO2ThrmCondChen.^3);
TK_CO2ThrmCondGupta=linspace(373,1348)';
CO2ThrmCondGupta=4.184*1e-3*(-2.400+2.16E-2*TK_CO2ThrmCondGupta-3.244E-
6*TK_CO2ThrmCondGupta.^2);
TK_CO2ThrmCondGoogle=[200;250;300;400;500;600;800;1000;1500;2000];
CO2ThrmCondGoogle=100*[1.05E-4;1.44E-4;1.81E-4;2.59E-4;3.33E-4;4.07E-
4;5.44E-4;6.65E-4;9.45E-4;1.18E-3];
TK_CO2ThrmCondCRC=[200;300;400;500;600];
CO2ThrmCondCRC=1e-3*[9.6;16.8;25.1;33.5;41.6];
TK_CO2ThrmCondEckert=[220;250;300;350;400;450;500;550;600];
CO2ThrmCondEckert=[0.010805;0.012884;0.016572;0.02047;0.02461;0.02897;0
.03352;0.03821;0.04311];
TK_CO2ThrmCondNBS=[180;190;200;210;220;230;240;250;260;270;280;290;300;
310;320;330;340;350;360;370;380;390;400;410;420;430;440;450;460;470;480
;490;500;510;520;530;540;550;560;570;580;590;600];
CO2ThrmCondNBS=0.01455*[0.567;0.610;0.659;0.699;0.743;0.791;0.837;0.886
;0.935;0.984;1.035;1.087;1.139;1.191;1.242;1.297;1.352;1.406;1.464;1.51
8;1.576;1.634;1.691;1.752;1.812;1.869;1.930;1.990;2.053;2.114;2.177;2.2
40;2.304;2.367;2.430;2.496;2.562;2.626;2.692;2.761;2.827;2.893;2.962];
plot(TK_CO2ThrmCondNBS,CO2ThrmCondNBS,'.k',TK_CO2ThrmCondGoogle,CO2Thrm
CondGoogle,'*g', TK_CO2ThrmCondGupta,CO2ThrmCondGupta,'--b',
TK_CO2ThrmCondChen,CO2ThrmCondChen,':r',
TK_CO2ThrmCondCRC,CO2ThrmCondCRC,'+r',TK_CO2ThrmCondEckert,CO2ThrmCondE
ckert,'*r'),
xlabel('Kelvins'),
ylabel('Watts meter^-^1 Kelvin^-^1'),
title('CO_2 Thermal Conductivity Data'),
legend('NBS data','Google data','Gupta data','Chen data','CRC
data','Eckert data','Location','SouthEast')
```



Next we approximate this data by using least squares techniques to fit cubic splines with a priori specified knot sequences.

```
knots=augknt([0 500 1000 1500 2000],4);
kth_CO2_pp=fn2fm(spap2(knots,4,[TK_CO2ThrmCondNBS;
TK_CO2ThrmCondGoogle; TK_CO2ThrmCondGupta; TK_CO2ThrmCondChen;
TK_CO2ThrmCondCRC; TK_CO2ThrmCondEckert], [CO2ThrmCondNBS;
CO2ThrmCondGoogle; CO2ThrmCondGupta; CO2ThrmCondChen; CO2ThrmCondCRC;
CO2ThrmCondEckert]),'pp');
kth_CO2_fit=fnval(kth_CO2_pp,linspace(000,2500,500));

plot(TK_CO2ThrmCondNBS,CO2ThrmCondNBS,'.k',TK_CO2ThrmCondGoogle,
CO2ThrmCondGoogle,'*g', TK_CO2ThrmCondGupta,CO2ThrmCondGupta,'--b',
TK_CO2ThrmCondChen,CO2ThrmCondChen,':r',TK_CO2ThrmCondCRC,
CO2ThrmCondCRC,'+r', TK_CO2ThrmCondEckert,CO2ThrmCondEckert,'*r',
linspace(000,2500,500), kth_CO2_fit,'-c'),
xlabel('Kelvins'),
ylabel('Watts meter^-^1 Kelvin^-^1'),
title('  Spline Fit to CO_2 Thermal Conductivity Data '),
legend('NBS data','Google data','Gupta data','Chen data','CRC
data','Eckert data','Spline fit','Location','SouthEast')
format long g,
display(kth_CO2_pp),
kth_CO2_pp.coefs(:,1),
kth_CO2_pp.coefs(:,2),
kth_CO2_pp.coefs(:,3),
kth_CO2_pp.coefs(:,4)

kth_CO2_pp =
```

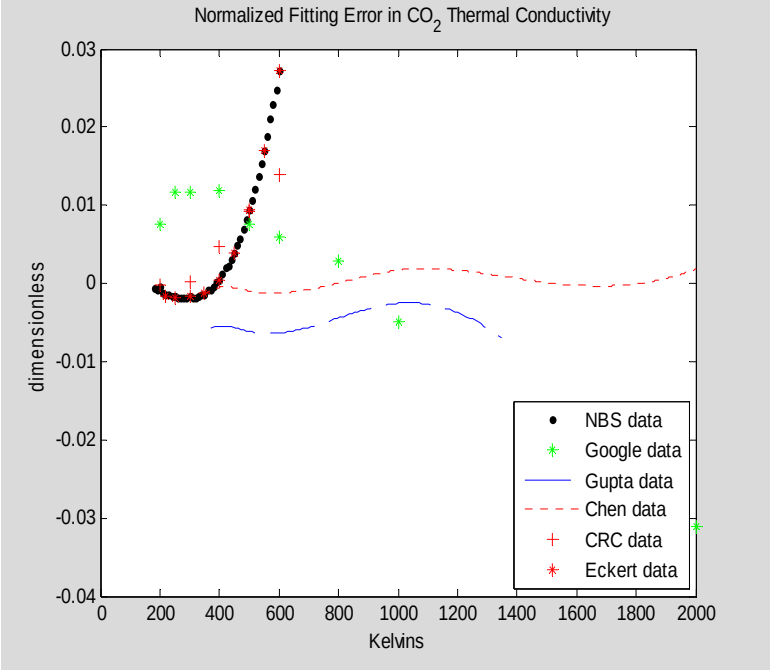
```
      form: 'pp'
    breaks: [0 500 1000 1500 2000]
     coefs: [4x4 double]
    pieces: 4
     order: 4
       dim: 1
ans =
    -8.95697919772388e-011
     8.7159435478886e-012
    -2.53405571179377e-013
     1.21190485215506e-012
ans =
     1.1296715060041e-007
    -2.13875373654485e-008
    -8.31362204361567e-009
    -8.69373040038474e-009
ans =
     3.19226886475612e-005
     7.77124952650418e-005
     6.28619155605098e-005
     5.43582393385096e-005
ans =
    -0.000558147134909062
       0.0324487608418191
        0.067047617076464
       0.0963684936494176
```



Then we examine the approximation error introduced by the spline fit.

```
CO2ThrmCondRange=max(CO2ThrmCondChen)-min(CO2ThrmCondNBS);
plot(TK_CO2ThrmCondNBS,(CO2ThrmCondNBS-
fnval(kth_CO2_pp,TK_CO2ThrmCondNBS))/CO2ThrmCondRange,'.k',
```

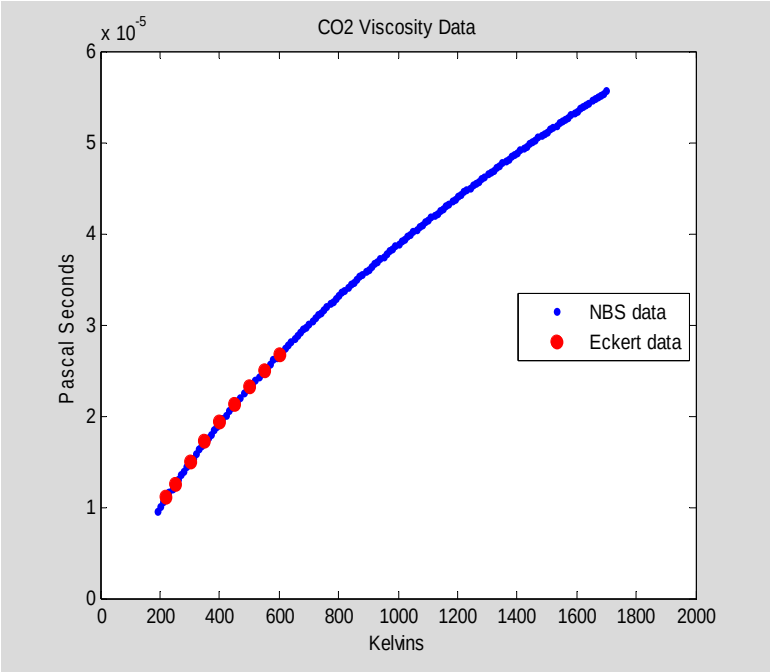
```
TK_CO2ThrmCondGoogle, (CO2ThrmCondGoogle-
fnval(kth_CO2_pp,TK_CO2ThrmCondGoogle))/CO2ThrmCondRange,'*g',
TK_CO2ThrmCondGupta,(CO2ThrmCondGupta-
fnval(kth_CO2_pp,TK_CO2ThrmCondGupta))/CO2ThrmCondRange,'--b',
TK_CO2ThrmCondChen,(CO2ThrmCondChen-
fnval(kth_CO2_pp,TK_CO2ThrmCondChen))/CO2ThrmCondRange,':r',
TK_CO2ThrmCondCRC, (CO2ThrmCondCRC-
fnval(kth_CO2_pp,TK_CO2ThrmCondCRC))/CO2ThrmCondRange,'+r',
TK_CO2ThrmCondEckert,(CO2ThrmCondEckert-
fnval(kth_CO2_pp,TK_CO2ThrmCondEckert))/CO2ThrmCondRange,'*r'),
xlabel('Kelvins'),
ylabel('dimensionless'),
title('  Normalized Fitting Error in CO_2 Thermal Conductivity'),
legend('NBS data','Google data','Gupta data','Chen data','CRC
data','Eckert data','Location','SouthEast')
```



C.4.6 Carbon Dioxide Gas Viscosity

Viscosity data for carbon dioxide gas was found from the CRC handbook and from NBS Circular 564 Values here are given in Pascal-seconds, and are tabulated versus temperature in Kelvins. Since the viscosities of gases such as carbon dioxide are almost independent of pressure in the range of pressures relevant to this work, no pressure dependence is modeled here.

```
TK_CO2ViscNBS=[190;200;210;220;230;240;250;260;270;280;290;300;310;320;
330;340;350;360;370;380;390;400;410;420;430;440;450;460;470;480;490;500
;510;520;530;540;550;560;570;580;590;600;610;620;630;640;650;660;670;68
0;690;700;710;720;730;740;750;760;770;780;790;800;810;820;830;840;850;8
60;870;880;890;900;910;920;930;940;950;960;970;980;990;1000;1010;1020;1
030;
1040;1050;1060;1070;1080;1090;1100;1110;1120;1130;1140;1150;1160;1170;
1180;1190;1200;1210;1220;1230;1240;1250;1260;1270;1280;1290;1300;1310;
1320;1330;1340;1350;1360;1370;1380;1390;1400;1410;1420;1430;1440;1450;
1460;1470;1480;1490;1500;1510;1520;1530;1540;1550;1560;1570;1580;1590;
1600;1610;1620;1630;1640;1650;1660;1670;1680; 1690;1700];
CO2ViscNBS=1.3701E-5*
[0.7002;0.7375;0.7745;0.8111;0.8474;0.8833;0.9189;0.9542;0.9891;
1.024;1.058;1.091;1.125;1.158;1.191;1.224;1.256;1.287;1.318;1.349;
1.380;
1.410;1.440;1.470;1.499;1.528;1.557;1.585;1.613;1.641;1.669;1.697;1.724
;
1.750;1.777;1.804;1.830;1.856;1.882;1.908;1.933;1.958;1.983;2.008;2.032
;
2.056;2.080;2.105;2.129;2.153;2.176;2.199;2.222;2.245;2.268;2.291;2.314
;
2.336;2.358;2.380;2.403;2.425;2.446;2.468;2.490;2.511;2.532;2.554;2.575
;
2.596;2.617;2.637;2.658;2.678;2.699;2.719;2.740;2.760;2.780;2.800;2.820
;
2.840;2.859;2.878;2.897;2.915;2.934;2.953;2.972;2.991;3.010;3.029;3.048
;
3.067;3.085;3.104;3.123;3.142;3.161;3.180;3.199;3.218;3.236;3.254;3.272
;
3.289;3.306;3.324;3.342;3.359;3.377;3.394;3.412;3.430;3.447;3.465;3.483
;
3.500;3.518;3.535;3.553;3.570;3.587;3.604;3.621;3.638;3.654;3.671;3.688
;
3.704;3.721;3.737;3.754;3.770;3.787;3.804;3.820;3.837;3.854;3.870;3.887
; 3.903;3.919;3.935;3.951;3.967;3.983;3.999;4.014;4.030;4.046;4.062];
TK_CO2ViscEckert=[220;250;300;350;400;450;500;550;600];CO2ViscEckert=1e
-6* [11.105;12.590;14.958;17.205;19.32;21.34;23.26;25.08;26.83];
plot(TK_CO2ViscNBS,CO2ViscNBS,'.b',TK_CO2ViscEckert,CO2ViscEckert,'or',
'MarkerFaceColor',[1 0 0]),
xlim([0 2000]),
xlabel('Kelvins'),
ylabel('Pascal Seconds'),
title('CO2 Viscosity Data'),
legend('NBS data','Eckert data','Location','East')
```

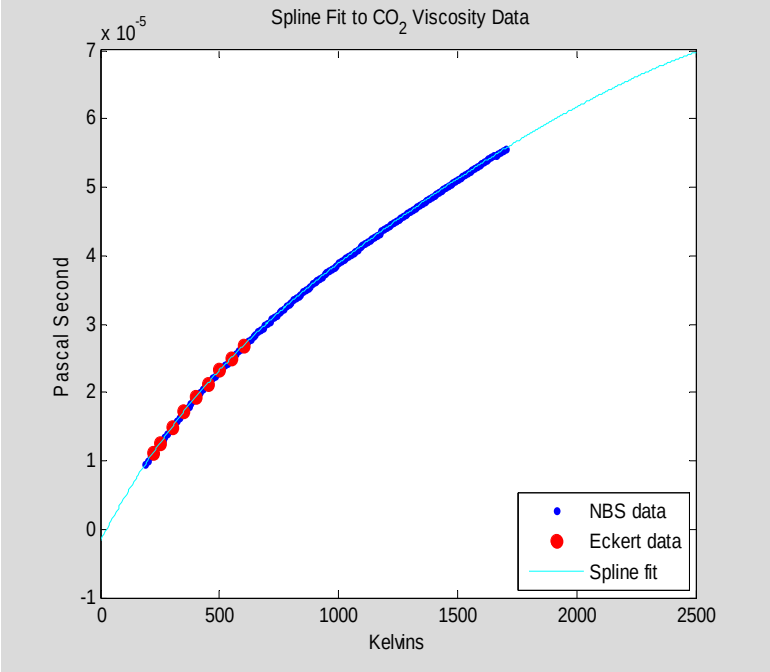


Next we approximate this data by using least squares techniques to fit a cubic spline with knot sequences specified a priori.

```
knots=augknt([0 500 1000 1500 2000],4);
mu_CO2_pp=fn2fm(spap2(knots,4,[TK_CO2ViscNBS; TK_CO2ViscEckert],
[CO2ViscNBS; CO2ViscEckert]),'pp');
mu_CO2_fit=fnval(mu_CO2_pp,linspace(000,2500,500));

plot(TK_CO2ViscNBS,CO2ViscNBS,'.b',
TK_CO2ViscEckert,CO2ViscEckert,'or', linspace(000,2500,500),
mu_CO2_fit,'-c','MarkerFaceColor','r'),
xlabel('Kelvins'),
ylabel('Pascal Second'),
title('  Spline Fit to CO_2 Viscosity Data '),
legend('NBS data','Eckert data','Spline fit','Location','SouthEast')
format long g,
display(mu_CO2_pp),
mu_CO2_pp.coefs(:,1),
mu_CO2_pp.coefs(:,2),
mu_CO2_pp.coefs(:,3),
mu_CO2_pp.coefs(:,4)
```

```
mu_CO2_pp =
      form: 'pp'
    breaks: [0 500 1000 1500 2000]
     coefs: [4x4 double]
    pieces: 4
     order: 4
       dim: 1
```

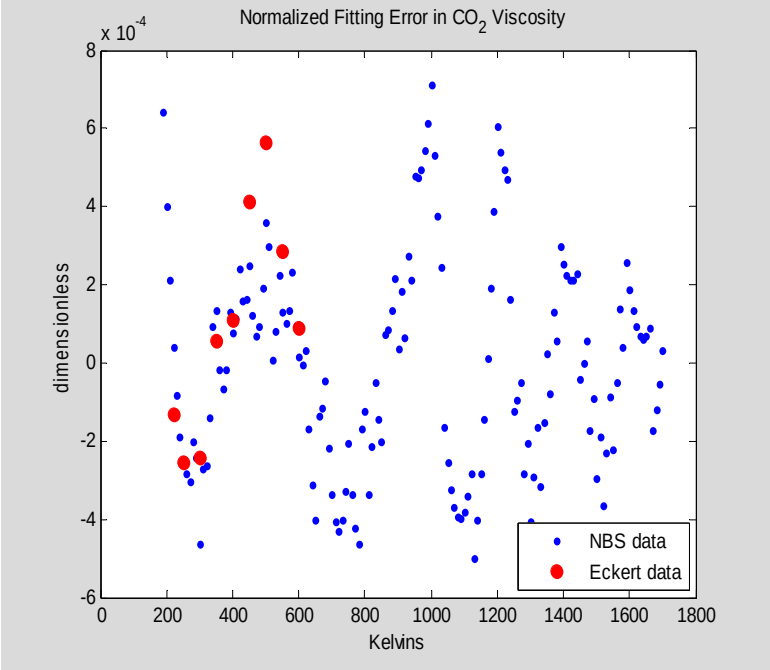
```
ans =
     1.84521269180779e-014
     7.03264658699861e-015
      1.5547749929767e-015
    -1.36947452041497e-015
ans =
    -4.34734084418281e-011
    -1.57952180647113e-011
    -5.24624818421333e-012
    -2.91408569474838e-012
ans =
     6.70617473216595e-008
     3.74274340683898e-008
     2.69067009439274e-008
     2.28265340044466e-008
ans =
    -1.73497359299849e-006
     2.32340638221339e-005
     3.88780571635258e-005
     5.12141924635583e-005
```



Next we examine the approximation error introduced by this spline fit.

```
CO2ViscRange=max(CO2ViscNBS)-min(CO2ViscNBS);
plot(TK_CO2ViscNBS,(CO2ViscNBS-
fnval(mu_CO2_pp,TK_CO2ViscNBS))/CO2ViscRange,'.b',
TK_CO2ViscEckert,(CO2ViscEckert-
fnval(mu_CO2_pp,TK_CO2ViscEckert))/CO2ViscRange,'or','MarkerFaceColor',
'r'),
xlabel('Kelvins'),
ylabel('dimensionless'),
title('  Normalized Fitting Error in CO_2 Viscosity'),
```

```
legend('NBS data','Eckert data','Location','SouthEast')
```



C.5 Helium Gas Properties

C.5.1 Helium Gas Density

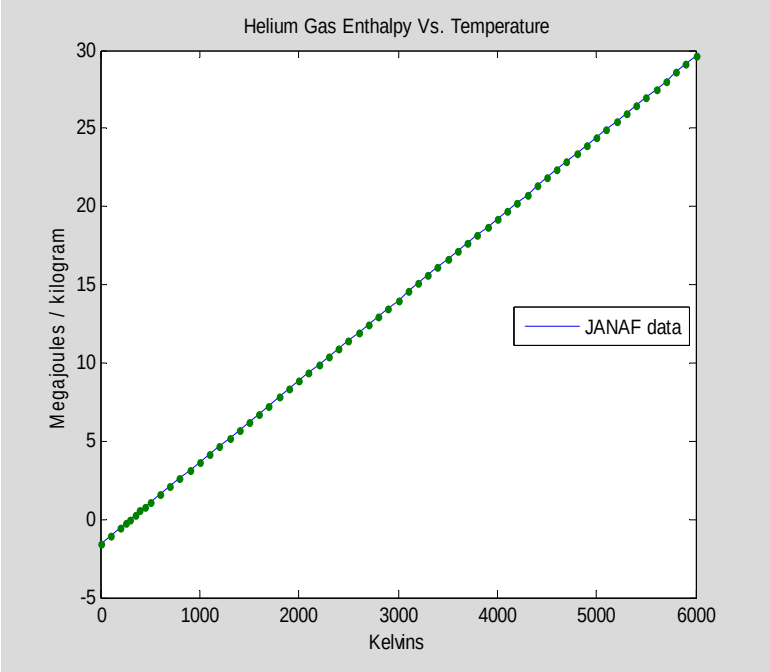
$$\rho = \frac{p}{RT} \text{ where}$$

$$R = \frac{\mathcal{R}}{\hat{M}} = \frac{8.31361 \text{ Joule/mole/Kelvin}}{0.00400260 \text{ kg/mole}} = 2077.0524 \text{ Joule kg}^{-1} \text{ Kelvin}^{-1}$$

C.5.2 Helium Gas Enthalpy

Helium enthalpy is taken from the published JANAF tables as follows. Enthalpy values here are given in kilojoules per mole and are tabulated vs temperatures in Kelvins. The mass of a mole of argon is 0.00400260 kilogram, a fact used to convert enthalpy to the megajoules per kilogram form plotted in the graph below.

```
TK_H0_He=[0;100;200;250;298.15;300;350;400;450;500;600;700;800;900;1000
;1100;1200;1300;1400;1500;1600;1700;1800;1900;2000;2100;2200;2300;2400;
2500;2600;2700;2800;2900;3000;3100;3200;3300;3400;3500;3600;3700;3800;
3900;4000;4100;4200;4300;4400;4500;4600;4700;4800;4900;5000;5100;5200;
5300;5400;5500;5600;5700;5800;5900;6000];
H0_He =[-6.197; -4.119; -2.040; -1.001; 0.; 0.038; 1.078; 2.117; 3.156;
4.196; 6.274; 8.353; 10.431; 12.510; 14.589; 16.667; 18.746; 20.824;
22.903; 24.982; 27.060; 29.139; 31.217; 33.296; 35.375; 37.453; 39.532;
41.610; 43.689; 45.768; 47.846; 49.925; 52.004; 54.082; 56.161; 58.239;
60.318; 62.397; 64.475; 66.554; 68.632; 70.711; 72.790; 74.868; 76.947;
79.025; 81.104; 83.183; 85.261; 87.340; 89.418; 91.497; 93.576; 95.654;
97.733; 99.811; 101.890; 103.969; 106.047; 108.126; 110.204; 112.283;
114.362; 116.440; 118.519];
plot(TK_H0_He,H0_He/0.0040026/1000,TK_H0_He,H0_He/0.0040026/1000,'.'),
xlabel('Kelvins'),
ylabel('Megajoules / kilogram'),
title('Helium Gas Enthalpy Vs. Temperature'),
legend('JANAF data','Location','East')
```

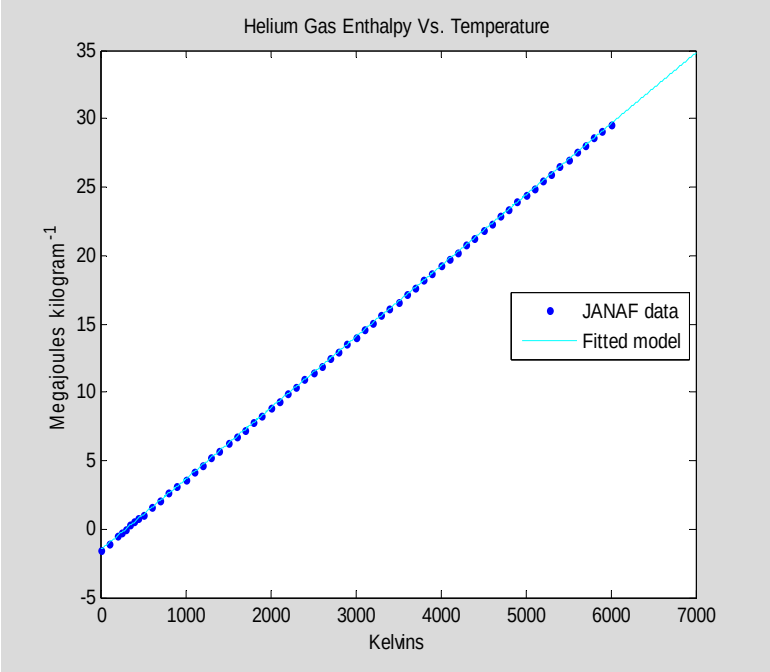


Next we approximate this data by using least squares techniques to fit cubic splines with a priori specified knot sequences.

```
knots=augknt([0 100 500 1000 1500 2000 3000 4000 5000 6000],4);
H0_He_pp=fn2fm(spap2(knots,4,TK_H0_He,H0_He/0.0040026*1000),'pp');
H0_He_fit=fnval(H0_He_pp,linspace(000,7000,500));
plot(TK_H0_He,H0_He/0.0040026/1000,'.',linspace(000,7000,500),
H0_He_fit/1e6,'-c'), xlabel('Kelvins'), ylabel('Megajoules kilogram^-
^1'), title('Helium Gas Enthalpy Vs. Temperature'), legend('JANAF
data','Fitted model','Location','East'),
format long g,
display(H0_He_pp),
H0_He_pp.coefs(:,1),
H0_He_pp.coefs(:,2),
H0_He_pp.coefs(:,3),
H0_He_pp.coefs(:,4)
```

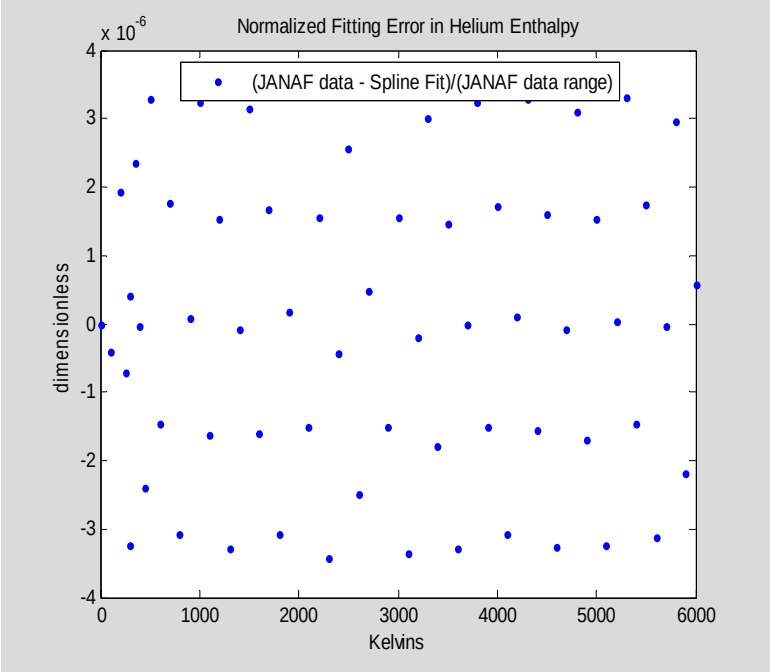
```
H0_He_pp =
      form: 'pp'
    breaks: [0 100 500 1000 1500 2000 3000 4000 5000 6000]
     coefs: [9x4 double]
    pieces: 9
     order: 4
       dim: 1
ans =
     -0.000186405607513449
     1.04775536214561e-006
     -8.3231321854934e-008
    -2.90484242341336e-008
```

```
     1.07466597637944e-007
    -6.36807356077043e-008
     4.16553583787997e-008
    -2.97813427846449e-008
     9.04949987564881e-008
ans =
        0.0547789931601892
      -0.00114268909384236
       0.000114617340728728
    -1.02296420591301e-005
     -5.3802278405783e-005
      0.000107397618046889
    -8.36445887816808e-005
     4.13214863683606e-005
     -4.8022541976934e-005
ans =
          5188.1380421324
         5193.50167253903
         5193.09044383779
         5193.14263768713
         5193.11062172689
         5193.13741939671
         5193.16117242599
         5193.11884932355
         5193.11214826793
ans =
        -1548243.64163294
        -1029068.45309561
         1048216.44200817
         3644779.91434701
         6241345.04472703
         8837900.33834558
         14031081.4746247
         19224200.6578203
         24417331.0472875
```



Then we examine the approximation error introduced by this spline fit.

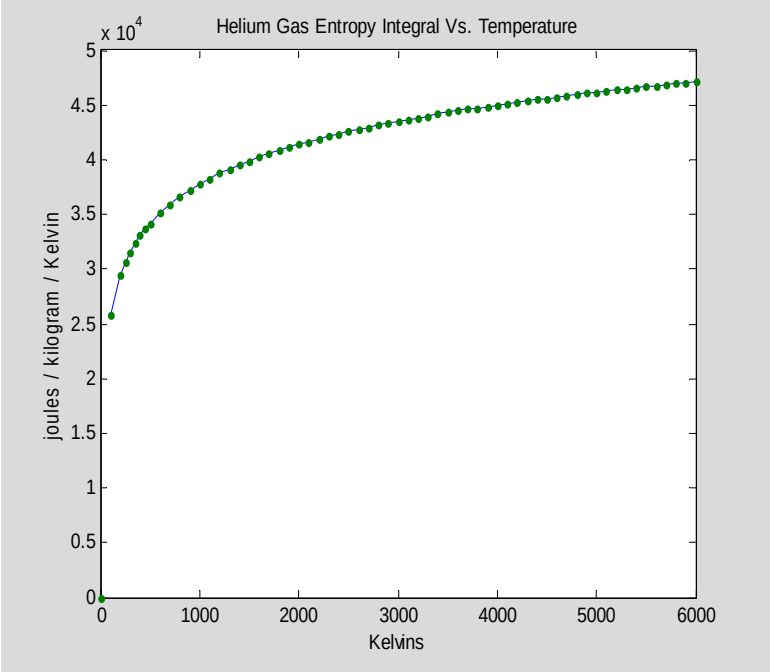
```
H0_He_fit=fnval(H0_He_pp,linspace(0,7000,500));
plot(TK_H0_He,(H0_He-0.0040026*fnval(H0_He_pp,TK_H0_He)/1000)
/(max(H0_He)-min(H0_He)),'.'),
xlabel('Kelvins'),
ylabel('dimensionless'), title('    Normalized Fitting Error in Helium
Enthalpy'), legend('(JANAF data - Spline Fit)/(JANAF data
range)','Location','North')
```



C.5.3 Helium Gas Entropy Integral

Helium entropy integral values are taken from the published JANAF tables as follows. Entropy integral values here are given in joules per mole per Kelvin and are tabulated vs temperature in Kelvins. The mass of a mole of argon is 0.0040026 kilogram, a fact used to convert enthalpy to the joules per kilogram per Kelvin form plotted in the graph below.

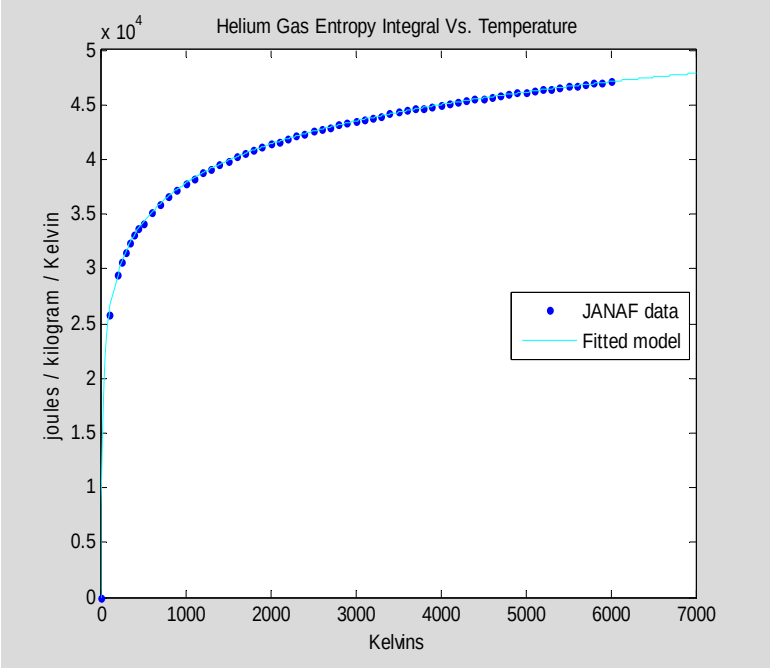
```
TK_S0_He=[0;100;200;250;298.15;300;350;400;450;500;600;700;800;900;1000
;1100;1200;1300;1400;1500;1600;1700;1800;1900;2000;2100;2200;2300;2400;
2500;2600;2700;2800;2900;3000;3100;3200;3300;3400;3500;3600;3700;3800;
3900;4000;4100;4200;4300;4400;4500;4600;4700;4800;4900;5000;5100;5200;
5300;5400;5500;5600;5700;5800;5900;6000];
S0_He =[0.; 103.445; 117.853; 122.491; 126.152; 126.281; 129.485;
132.260; 134.709; 136.899; 140.688; 143.893; 146.868; 149.116; 151.306;
153.288; 155.096; 156.760; 158.300; 159.734; 161.076; 162.336; 163.524;
164.648; 165.714; 166.728; 167.695; 168.619; 169.504; 170.352; 171.168;
171.952; 172.708; 173.438; 174.142; 174.824; 175.484; 176.123; 176.744;
177.346; 177.932; 178.501; 179.056; 179.596; 180.122; 180.635; 181.136;
181.625; 182.103; 182.570; 183.027; 183.474; 183.912; 184.340; 184.760;
185.172; 185.575; 185.971; 186.360; 186.741; 187.116; 187.484; 187.845;
188.201; 188.550];
plot(TK_S0_He(2:end),S0_He(2:end)/0.0040026,TK_S0_He,S0_He/0.0040026,'.
'), xlabel('Kelvins'),
ylabel('joules / kilogram / Kelvin'),
title('Helium Gas Entropy Integral Vs. Temperature')
```



Next, we develop a leasst-squares spline fit to this data using a fixed knot sequence.

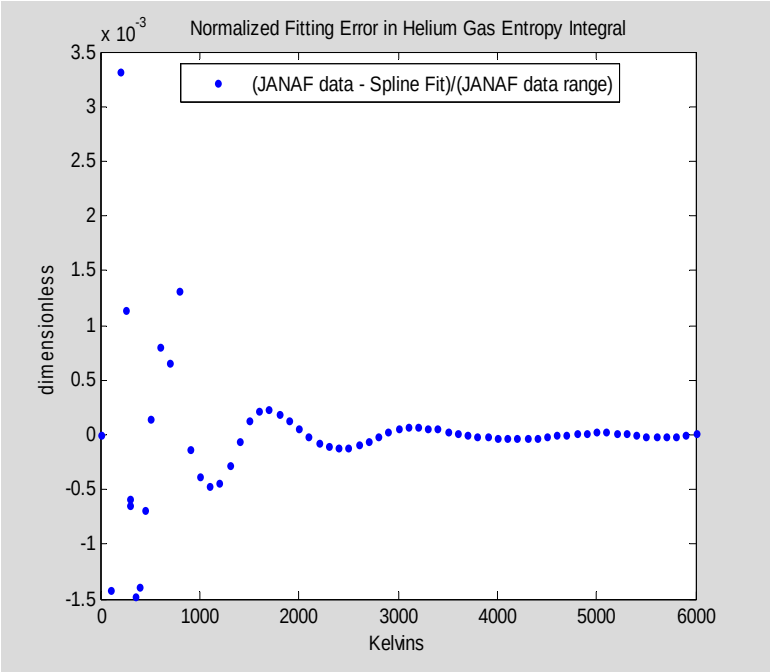
```
knots=augknt([0 100 500 1000 1500 2000 3000 4000 5000 6000],4);
S0_He_pp=fn2fm(spap2(knots,4,TK_S0_He,S0_He/0.0040026),'pp');
S0_He_fit=fnval(S0_He_pp,linspace(0,7000,500));
plot(TK_S0_He,S0_He/0.0040026,'.',linspace(0,7000,500), S0_He_fit,'-
c'), xlabel('Kelvins'), ylabel('joules / kilogram / Kelvin'),
title('Helium Gas Entropy Integral Vs. Temperature'), legend('JANAF
data','Fitted model','Location','East'),
format long g,
display(S0_He_pp),
S0_He_pp.coefs(:,1),
S0_He_pp.coefs(:,2),
S0_He_pp.coefs(:,3),
S0_He_pp.coefs(:,4)

S0_He_pp =
      form: 'pp'
    breaks: [0 100 500 1000 1500 2000 3000 4000 5000 6000]
     coefs: [9x4 double]
    pieces: 9
     order: 4
       dim: 1
ans =
        0.0211828025756756
     5.35643995955314e-005
     2.07048677446397e-006
     1.36481835840593e-006
     1.30570089689302e-007
     1.38984183639296e-007
     2.51291660107096e-008
```

```
     2.69436523468121e-008
    -4.2066915216132e-010
ans =
         -6.42513085801957
        -0.0702900853168898
        -0.0060128058022521
       -0.00290707564055624
      -0.000859848102947319
       -0.00066399296841336
      -0.000247040417495464
      -0.000171652919463337
     -9.08219624229167e-005
ans =
           689.799279192033
           40.2571848583877
           9.73602841073088
            5.2760876893268
           3.39262581757498
           2.63070528189464
            1.7196718959858
             1.300978559027
           1.03850367714076
ans =
                          0
           25911.4219146833
             34196.00378145
           37819.6273830604
           39901.5046123855
           41399.1767566473
           43504.8732537679
           45002.6338982689
           46158.9031901794
```



Then we examine the residual discrepancies introduced by the spline fit.

```
S0_He_fit=fnval(S0_He_pp,linspace(0,7000,500)); plot(TK_S0_He,(S0_He-
0.0040026*fnval(S0_He_pp,TK_S0_He)) /(max(S0_He)-min(S0_He)),'.'),
xlabel('Kelvins'),
ylabel('dimensionless'), title('    Normalized Fitting Error in Helium
Gas Entropy Integral'), legend('(JANAF data - Spline Fit)/(JANAF data
range)','Location','North')
```

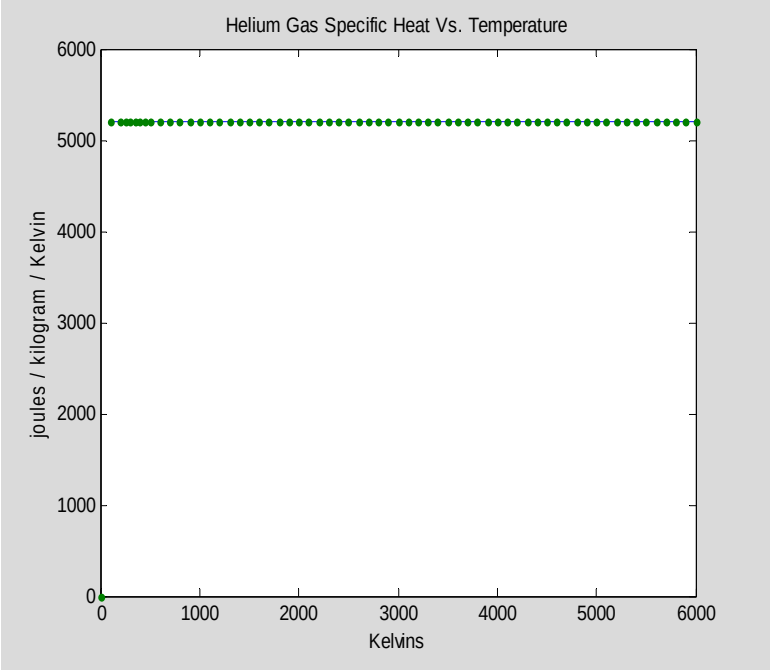


C.5.4 Helium Gas Specific Heat

Helium gas specific heat at constant pressure is taken from the published JANAF tables as follows. Specific heat values here are given in joules per mole per Kelvin and are tabulated vs temperature in Kelvins. The mass of a mole of helium is 0.0040026 kilogram, a fact used to convert specific heat to the joules per kilogram per Kelvin form plotted in the graph below.

```
TK_Cp_He=[0;100;200;250;298.15;300;350;400;450;500;600;700;800;900;1000
;1100;1200;1300;1400;1500;1600;1700;1800;1900;2000;2100;2200;2300;2400;
2500;2600;2700;2800;2900;3000;3100;3200;3300;3400;3500;3600;3700;3800;
3900;4000;4100;4200;4300;4400;4500;4600;4700;4800;4900;5000;5100;5200;
5300;5400;5500;5600;5700;5800;5900;6000];
Cp_He=[0;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786
;
20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;
```

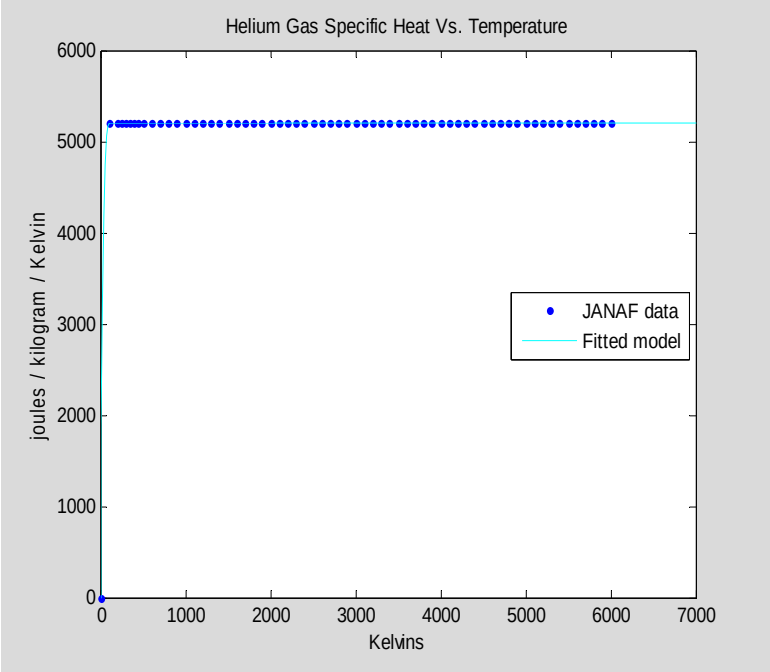
```
20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;
20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;
20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;
20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;20.786;
20.786;20.786;20.786;20.786;20.786];
plot(TK_Cp_He(2:end),Cp_He(2:end)/0.0040026,TK_Cp_He,Cp_He/0.0040026,'.
'),xlabel('Kelvins'),
ylabel('joules / kilogram / Kelvin'),
title('Helium Gas Specific Heat Vs. Temperature')
```



Next, we generate a spline approximation model fitted to the data.

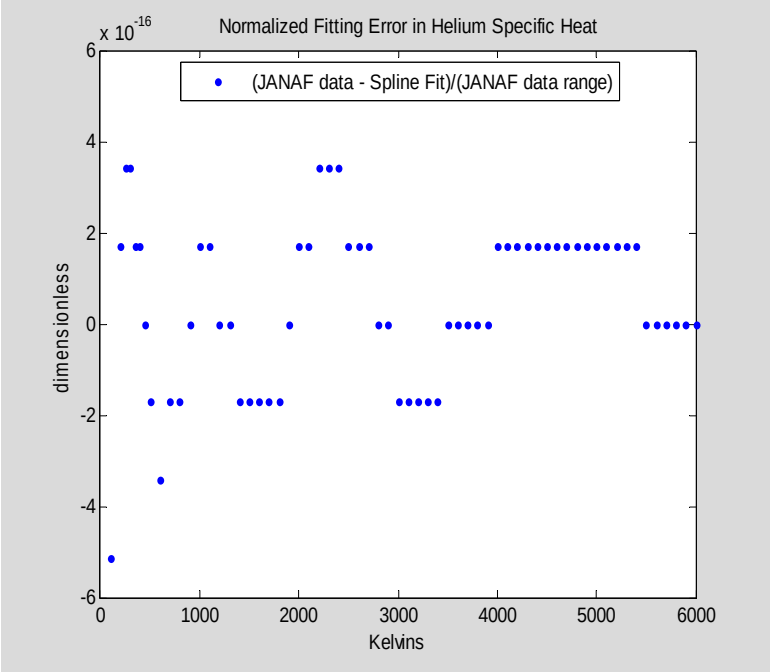
```
knots=augknt([0 100 500 1000 1500 2000 3000 4000 5000 6000],4);
Cp_He_pp=fn2fm(spap2(knots,4,TK_Cp_He,Cp_He/0.0040026),'pp');
Cp_He_fit=fnval(Cp_He_pp,linspace(000,7000,500));
plot(TK_Cp_He,Cp_He/0.0040026,'.',linspace(000,7000,500), Cp_He_fit,'-
c'), xlabel('Kelvins'), ylabel('joules / kilogram / Kelvin'),
title('Helium Gas Specific Heat Vs. Temperature'), legend('JANAF
data','Fitted model','Location','East'),
format long g,
display(Cp_He_pp),
Cp_He_pp.coefs(:,1),
Cp_He_pp.coefs(:,2),
Cp_He_pp.coefs(:,3)
Cp_He_pp.coefs(:,4)    ,

Cp_He_pp =
      form: 'pp'
    breaks: [0 100 500 1000 1500 2000 3000 4000 5000 6000]
     coefs: [9x4 double]
```

```
    pieces: 9
     order: 4
       dim: 1
ans =
        0.0051931244690951
    -2.19353439977862e-019
     5.45696821063757e-020
    -2.72848410531878e-020
     2.58700715170966e-020
     -5.6843418860808e-021
     1.36424205265939e-021
     2.27373675443232e-022
    -1.81898940354586e-021
ans =
         -1.55793734072853
     2.12215430413683e-016
    -5.45696821063757e-017
     2.18278728425503e-017
    -2.91038304567337e-017
     1.22781784739345e-017
    -2.72848410531878e-018
                         0
     2.72848410531878e-018
ans =
          155.793734072853
    -5.45696821063757e-014
     1.09139364212751e-014
                         0
     5.45696821063757e-015
    -5.45696821063757e-015
                         0
                         0
                         0
ans =
                         0
          5193.12446909509
          5193.12446909509
          5193.12446909509
          5193.12446909509
          5193.12446909509
          5193.12446909509
          5193.12446909509
          5193.12446909509
```



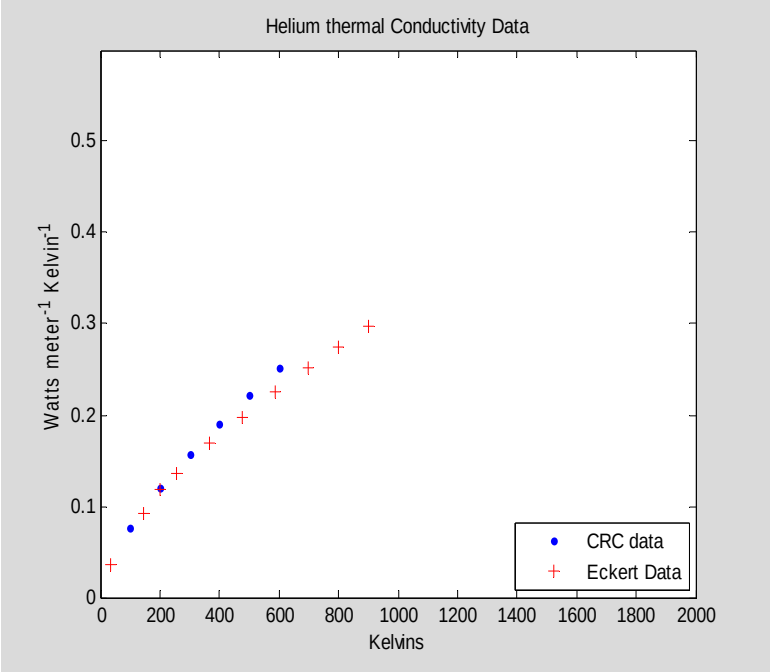
Then we examine the approximation error introduced by the spline fit.

```
Cp_He_fit=fnval(Cp_He_pp,linspace(0,7000,500)); plot(TK_Cp_He,(Cp_He-
0.0040026*fnval(Cp_He_pp,TK_Cp_He)) /(max(Cp_He)-min(Cp_He)),'.'),
xlabel('Kelvins'),
ylabel('dimensionless'), title('    Normalized Fitting Error in Helium
Specific Heat'), legend('(JANAF data - Spline Fit)/(JANAF data
range)','Location','North')
```



C.5.5 Helium Gas Thermal Conductivity

```
TK_HeThrmCondCRC1=[100;200;300;400;500;600];
HeThrmCondCRC1=1e-3* [75.5; 119.3; 156.7; 190.6; 222.3; 252.4];
TK_HeThrmCondEckert=[33; 144; 200; 255; 366; 477; 589; 700; 800; 900];
HeThrmCondEckert=[0.0353; 0.0928; 0.1177; 0.1357; 0.1691; 0.197; 0.225;
0.251; 0.275; 0.298];
plot(TK_HeThrmCondCRC1,HeThrmCondCRC1,'.b',TK_HeThrmCondEckert,HeThrmCo
ndEckert,'+r'),
xlabel('Kelvins'),
xlim([0 2000]),
ylim([0 0.6])
ylabel('Watts meter^-^1 Kelvin^-^1')
title('Helium thermal Conductivity Data'),
legend('CRC data','Eckert Data','Location','SouthEast')
```



Because of the paucity of helium thermal conductivity data found and its limited temperature range, additional information was sought. An approximate thermal conductivity algebraic formula based on modeling molecules as rigid spheres (called the *Sutherland model*) is presented in [Eckert and Drake 1972, 64-65], along with the disclaimer that it should not be relied upon over large temperature ranges. This formula is as follows:

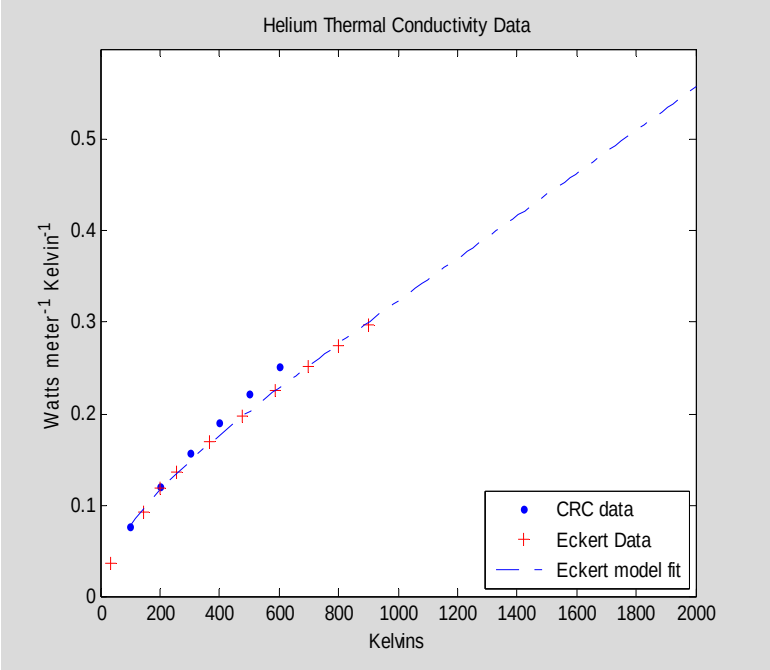
$$k = A\frac{(1+BT)T^{\frac{3}{2}}}{(C+T)}$$

where *A,B,C* are parameters to be fit to experimental data. Therefore, a least-squares fit of this model function to the Eckert data was made. Tenperature was also as a data weighting factor since the intended extrapolation is to higher temperatures. The following fitted parameter values resulted.
A=0.008117
B=2.723e-4
C=8.768

This fitted formula is plotted below along with the published data values.

```
A=0.008117;
B=2.723e-4;
C=8.768;
TK_HeThrmCondFormula=linspace(100,2000)';
HeThrmCondFormula=A*(1+B*TK_HeThrmCondFormula)
./(C+TK_HeThrmCondFormula).*TK_HeThrmCondFormula.^1.5;
```

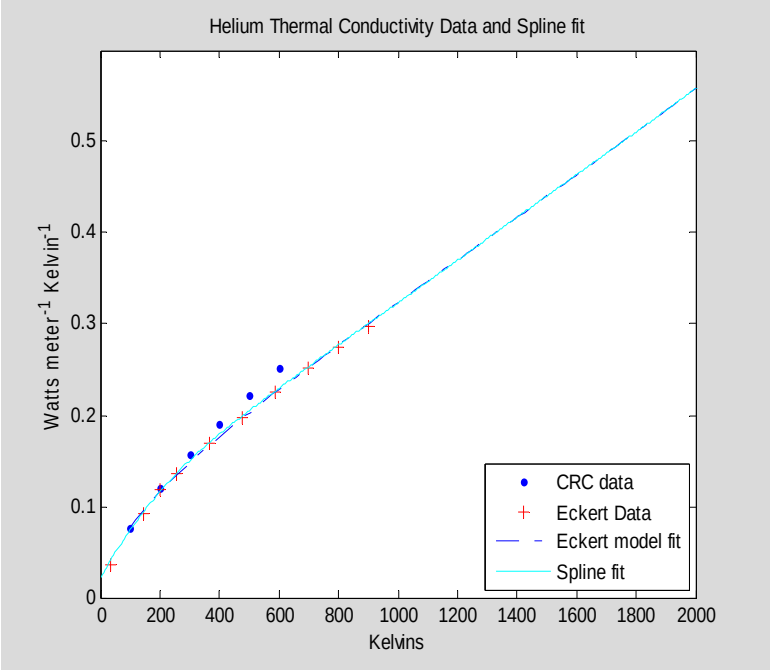
```
plot(TK_HeThrmCondCRC1,HeThrmCondCRC1,'.b',TK_HeThrmCondEckert,HeThrmCo
ndEckert,'+r',TK_HeThrmCondFormula,HeThrmCondFormula,'-.'),
xlabel('Kelvins'),
xlim([0 2000]),
ylim([0 0.6])
ylabel('Watts meter^-^1 Kelvin^-^1')
title('Helium Thermal Conductivity Data'),
legend('CRC data','Eckert Data','Eckert model
fit','Location','SouthEast')
```



Next we approximate this data by using least squares techniques to fit cubic splines with a priori specified knot sequences.

```
knots=augknt([0 500 1000 1500 2000],4);
kth_He_pp=fn2fm(spap2(knots,4,[TK_HeThrmCondEckert;  TK_HeThrmCondCRC1;
TK_HeThrmCondFormula], [HeThrmCondEckert;  HeThrmCondCRC1;
HeThrmCondFormula]),'pp');
kth_Hefit=fnval(kth_He_pp,linspace(000,2500,500));
plot(TK_HeThrmCondCRC1,HeThrmCondCRC1,'.b',TK_HeThrmCondEckert,HeThrmCo
ndEckert,'+r',TK_HeThrmCondFormula,HeThrmCondFormula,'-
.',linspace(000,2500,500),kth_Hefit,'-c'),
xlabel('Kelvins'),
xlim([0 2000]),
ylim([0 0.6])
ylabel('Watts meter^-^1 Kelvin^-^1')
title('Helium Thermal Conductivity Data and Spline fit'),
legend('CRC data','Eckert Data','Eckert model fit','Spline
fit','Location','SouthEast')
kth_He_pp.coefs(:,1),
kth_He_pp.coefs(:,2),
kth_He_pp.coefs(:,3),
```

```
kth_He_pp.coefs(:,4)

ans =
     3.92617328131614e-010
     2.98637889870604e-011
    -6.69940393922385e-012
      1.8145560550439e-011
ans =
    -6.27680083304969e-007
    -3.87540911075476e-008
     6.04159237304301e-009
    -4.00751353579309e-009
ans =
       0.00058211017104374
      0.000248893083837481
      0.000232536834470229
      0.000233553873888854
ans =
        0.0217316847044791
         0.204943915416558
         0.323434908181795
         0.440376298017767
```

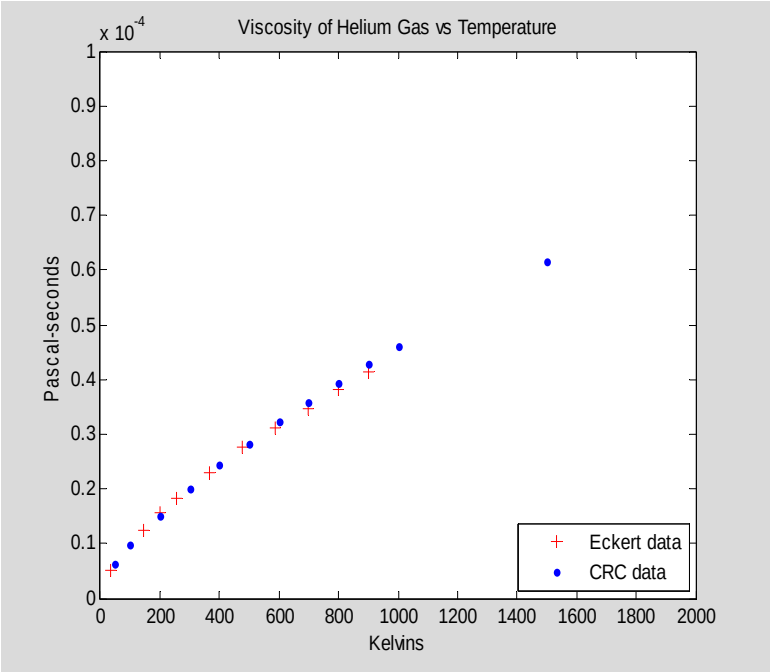


C.5.6 Helium Gas Viscosity

Viscosity data for helium gas was found in the CRC Handbook [Lide ed 1999, 6-19], and in [Eckert and Drake 1972, 780] Values here are given in Pascal-seconds, and are

tabulated versus temperature in Kelvins. Since the viscosities of gases such as helium are almost independent of pressure in the range of pressures relevant to this work, no pressure dependence is modeled here.

```
TK_HeViscEckert=[33; 144; 200; 255; 366; 477; 589; 700; 800; 900];
HeViscEckert=[50.2e-7; 125.5e-7; 156.6e-7; 181.7e-7; 230.5e-7; 275.0e-
7; 311.3e-7; 347.5e-7; 381.7e-7; 413.6e-7];
TK_HeViscCRC=[50; 100; 200; 300; 400; 500; 600; 700; 800; 900; 1000;
1500];
HeViscCRC=1e-6* [6.36; 9.78; 15.14; 19.93; 24.29; 28.36; 32.22; 35.89;
39.43;42.85; 46.16; 61.55];
plot(TK_HeViscEckert,HeViscEckert,'+r',TK_HeViscCRC,HeViscCRC,'.b'),
xlim([0 2000])
ylim([0 1e-4])
title('Viscosity of Helium Gas vs Temperature'),
ylabel('Pascal-seconds'),
xlabel('Kelvins'),
legend('Eckert data','CRC data','Location','SouthEast')
```



A simple algebraic formula for viscosity, also based on the Sutherland rigid sphere model of gas molecules, is offered by [Eckert and Drake 1972, 65], as follows:

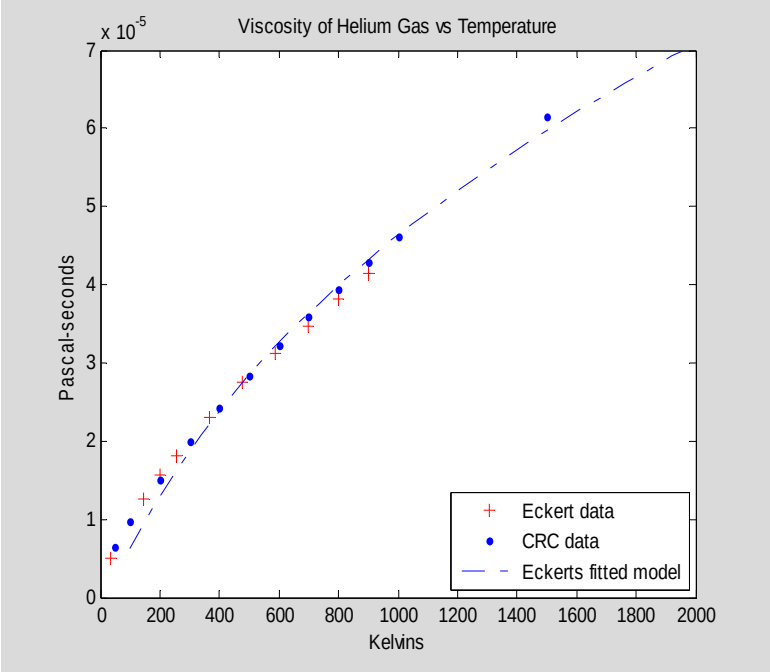
$$\mu = \frac{\mu_0 \sqrt{T}}{1 + E/T}$$

A T-weighted least-squares fit to the data plotted above yields the following model constants:

$$\mu = \frac{(1.224e-6)\sqrt{T}}{1 + (0.8221)/T}$$

this function is plotted below along with the data.

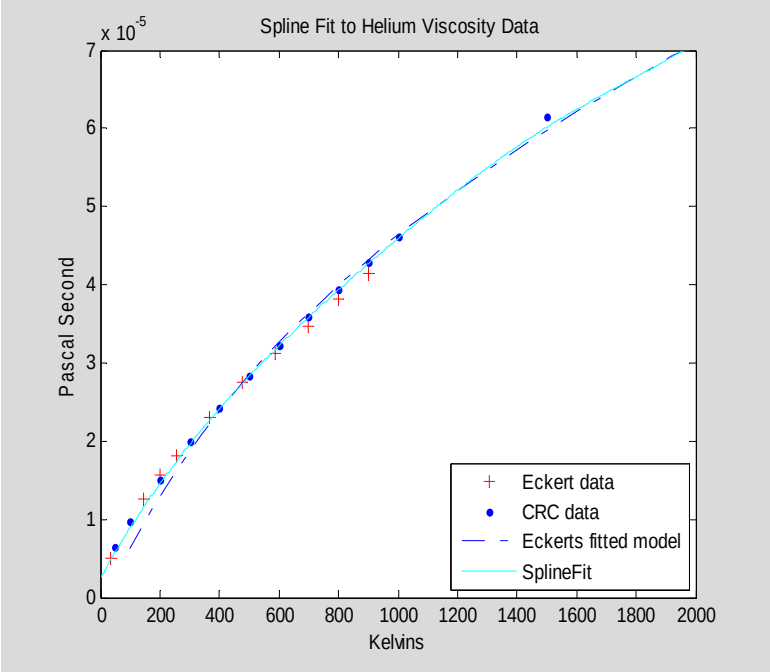
```
TK_ViscEckertform=linspace(100,2000,25)';
ViscEckertform=(1.7259e-6)*TK_ViscEckertform.^1.5
./(TK_ViscEckertform+177.9);
plot(TK_HeViscEckert,HeViscEckert,'+r',TK_HeViscCRC,HeViscCRC,'.b',TK_V
iscEckertform,ViscEckertform,'-.'),
xlim([0 2000])
ylim([0 0.7e-4])
title('Viscosity of Helium Gas vs Temperature'),
ylabel('Pascal-seconds'),
xlabel('Kelvins'),
legend('Eckert data','CRC data','Eckerts fitted
model','Location','SouthEast')
```



Next we approximate this data by using least squares techniques to fit a cubic spline with knot sequences specified a priori.

```
knots=augknt([0 500 1000 1500 2000],4);
mu_He_pp=fn2fm(spap2(knots,4,[TK_HeViscEckert;TK_HeViscCRC;TK_ViscEcker
tform],[HeViscEckert;HeViscCRC;ViscEckertform]),'pp');
mu_He_fit=fnval(mu_He_pp,linspace(000,2500,500)');
plot(TK_HeViscEckert,HeViscEckert,'+r',TK_HeViscCRC,HeViscCRC,'.b',TK_V
iscEckertform,ViscEckertform,'-.',linspace(000,2500,500)',mu_He_fit,'-
c'),
xlim([0 2000])
ylim([0 0.7e-4])
title('Viscosity of Helium Gas vs Temperature, with SplineFit'),
ylabel('Pascal-seconds'),
xlabel('Kelvins'),
```

```
legend('Eckert data','CRC data','Eckerts fitted
model','SplineFit','Location','SouthEast')
xlabel('Kelvins'),
ylabel('Pascal Second'),
title('  Spline Fit to Helium Viscosity Data '),
format long g,
display(mu_He_pp),
mu_He_pp.coefs(:,1),
mu_He_pp.coefs(:,2),
mu_He_pp.coefs(:,3),
mu_He_pp.coefs(:,4)

mu_He_pp =
      form: 'pp'
    breaks: [0 500 1000 1500 2000]
     coefs: [4x4 double]
    pieces: 4
     order: 4
       dim: 1
ans =
     1.91695479899686e-014
     6.24463127005045e-015
     -6.0365993852115e-015
     1.70236468954153e-014
ans =
    -4.17821486162275e-011
    -1.30278266312746e-011
    -3.66087972619893e-012
    -1.27157788040163e-011
ans =
     6.76064817179773e-008
     4.02014940942262e-008
     3.18571409154894e-008
     2.36688116503818e-008
ans =
     2.53036583099191e-006
     2.82842630346697e-005
     4.59086323327205e-005
      6.0167407935764e-005
```



C.6 Nitrogen Gas Properties

C.6.1 Nitrogen Gas Density

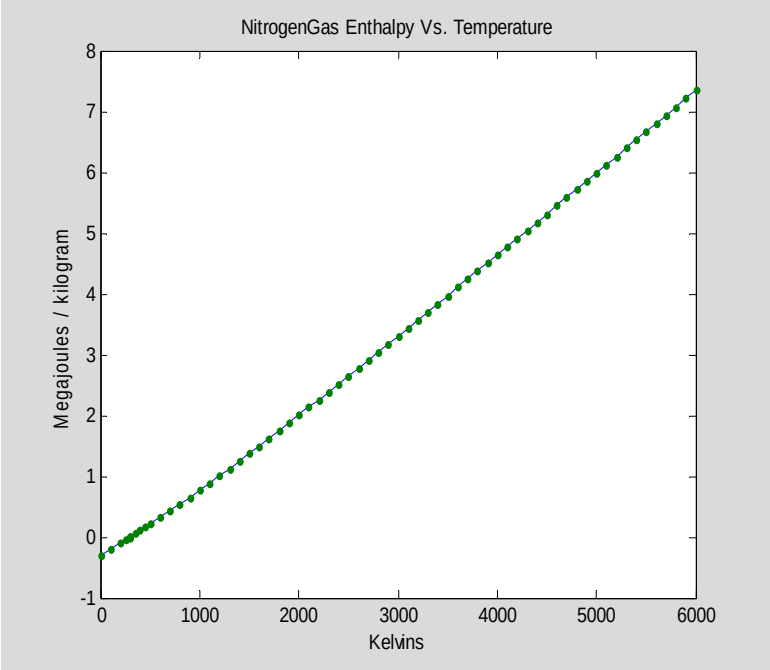
$$\rho = \frac{p}{RT} \text{ where } R = \frac{\mathcal{R}}{\hat{M}} = \frac{8.31361 \text{ Joule/mole/Kelvin}}{0.0280134 \text{ kg/mole}} = 296.7726 \text{ Joule kg}^{-1} \text{ Kelvin}^{-1}$$

C.6.2 Nitrogen Gas Enthalpy

Nitrogen enthalpy is taken from the published JANAF tables as follows. Enthalpy values here are given in kilojoules per mole and are tabulated vs temperatures in Kelvins. The mass of a mole of nitrogen is 0.0280134 kilogram, a fact used to convert enthalpy to the megajoules per kilogram form plotted in the graph below.

```
TK_H0_N2=[0;100;200;250;298.15;300;350;400;450;500;600;700;800;900;1000
;1100;1200;1300;1400;1500;1600;1700;1800;1900;2000;2100;2200;2300;2400;
2500;2600;2700;2800;2900;3000;3100;3200;3300;3400;3500;3600;3700;3800;3
900;4000;4100;4200;4300;4400;4500;4600;4700;4800;4900;5000;5100;5200;53
00;5400;5500;5600;5700;5800;5900;6000];
H0_N2 =[-8.67;-5.768;-2.857;-1.402;0;0.054;1.511;2.971;4.437;5.911;
8.894;11.937;15.046;18.223;21.463;24.76;28.109;31.503;34.936;38.405;
41.904;45.429;48.978;52.548;56.137;59.742;63.361;66.995;70.64;74.296;77
.963;
81.639;85.323;89.015;92.715;96.421;100.134;103.852;107.577;111.306;115.
```

```
041;
118.781;122.525;126.274;130.027;133.784;137.545;141.309;145.078;148.85;
152.625;156.405;160.187;163.973;167.763;171.556;175.352;179.152;182.955
;186.761;190.571;194.384;198.201;202.023;205.848];
plot(TK_H0_N2,H0_N2/0.0280134/1000,TK_H0_N2,H0_N2/0.0280134/1000,'.'),
xlabel('Kelvins'),
ylabel('Megajoules / kilogram'),
title('NitrogenGas Enthalpy Vs. Temperature')
```

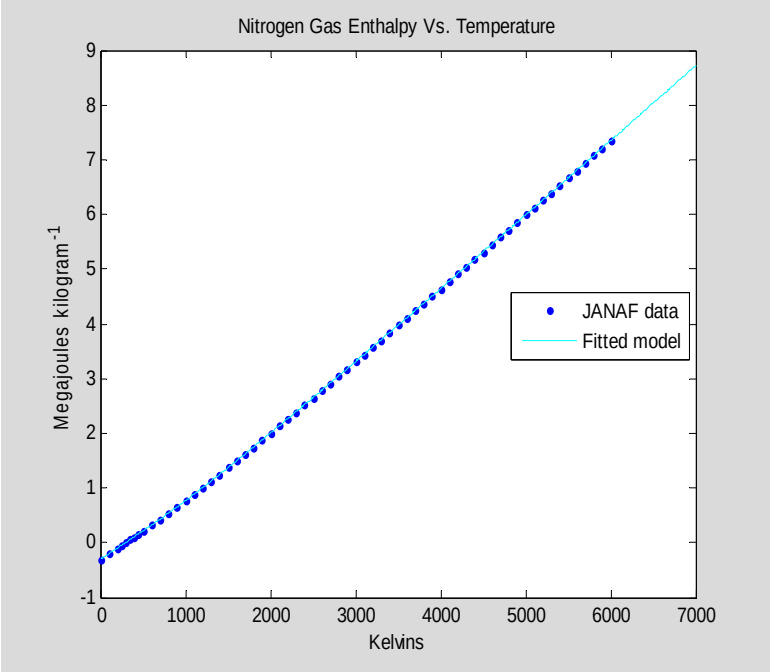


Next we approximate this data by using least squares techniques to fit cubic splines with a priori specified knot sequences.

```
knots=augknt([0 100 500 1000 1500 2000 3000 4000 5000 6000],4);
H0_N2_pp=fn2fm(spap2(knots,4,TK_H0_N2,H0_N2/0.0280134*1000),'pp');
H0_N2_fit=fnval(H0_N2_pp,linspace(000,7000,500));
plot(TK_H0_N2,H0_N2/0.0280134/1000,'.',linspace(000,7000,500),H0_N2_fit
/1e6,    '-c'),
xlabel('Kelvins'),
ylabel('Megajoules kilogram^-^1'),
title('Nitrogen Gas Enthalpy Vs. Temperature'),
legend('JANAF data','Fitted model','Location','East'),
format long g,
display(H0_N2_pp),
H0_N2_pp.coefs(:,1),
H0_N2_pp.coefs(:,2),
H0_N2_pp.coefs(:,3),
H0_N2_pp.coefs(:,4)
```

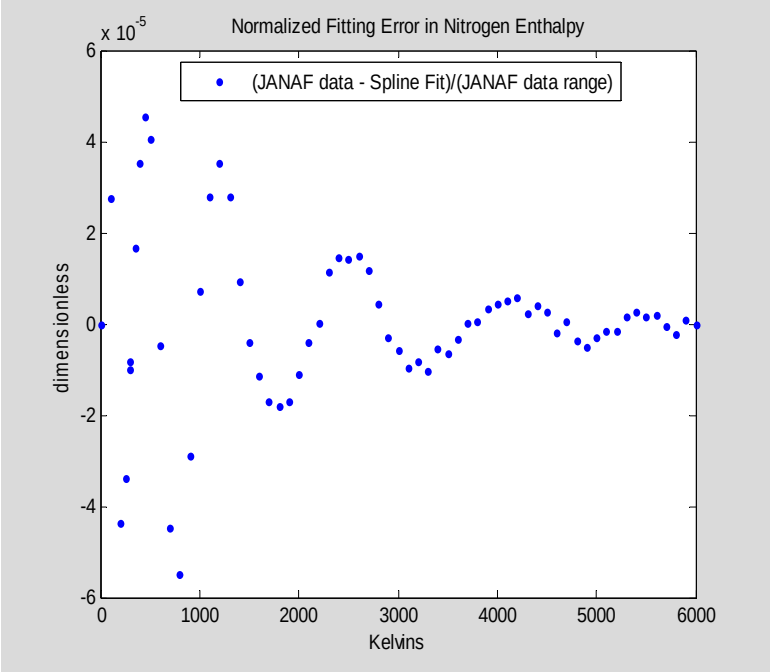
```
H0_N2_pp =
      form: 'pp'
```

```
    breaks: [0 100 500 1000 1500 2000 3000 4000 5000 6000]
     coefs: [9x4 double]
    pieces: 9
     order: 4
       dim: 1
ans =
     -0.00269638862197063
      0.000160833720360164
     3.88413102314841e-006
     -3.9770887585245e-005
    -1.50168106641709e-005
    -5.58052792502466e-006
     -1.1879336664623e-006
    -5.61037574534112e-007
     5.39105331688233e-007
ans =
        0.719899180225285
       -0.0890174063659046
        0.103983058066292
        0.109809254601013
       0.0501529232231483
       0.0276277072268938
       0.0108861234518217
      0.00732232245243768
      0.00563920972883602
ans =
         988.801689048395
         1051.88986643433
         1057.87612711449
         1164.77228344814
         1244.75337236022
         1283.64368758524
         1322.15751826395
         1340.36596416821
         1353.32749634948
ans =
        -309494.741802138
        -206111.969717016
         210694.549941223
         766113.894392933
          1370980.9888191
         2004018.80447198
         3309709.67135908
          4641565.3794084
         5988692.62845451
```



Then we examine the approximation error introduced by this spline fit.

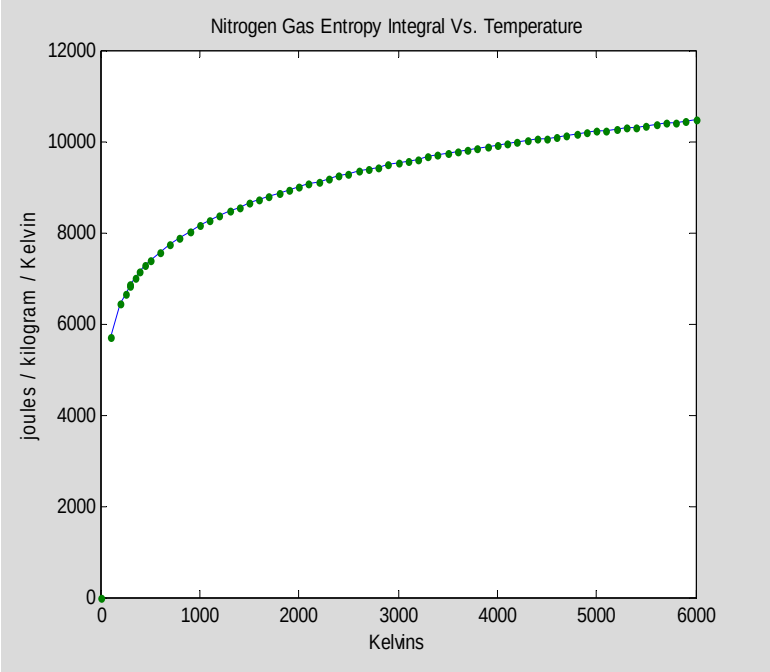
```
H0_N2_fit=fnval(H0_N2_pp,linspace(0,7000,500));
plot(TK_H0_N2,(H0_N2-0.0280134*fnval(H0_N2_pp,TK_H0_N2)/1000)
/(max(H0_N2)-min(H0_N2)),'.'),
xlabel('Kelvins'),
ylabel('dimensionless'), title('    Normalized Fitting Error in
Nitrogen Enthalpy'), legend('(JANAF data - Spline Fit)/(JANAF data
range)','Location','North')
```


Normalized Fitting Error in Nitrogen Enthalpy
x 10^-5
(JANAF data - Spline Fit)/(JANAF data range)
dimensionless
Kelvins
6
4
2
0
-2
-4
-6
0
1000
2000
3000
4000
5000
6000

C.6.3 Nitrogen Gas Entropy Integral

Nitrogen entropy integral values are taken from the published JANAF tables as follows. Entropy values here were given in joules per mole per Kelvin and were tabulated vs temperature in Kelvins. The mass of a mole of argon is 0.0280134 kilogram, a fact used to convert enthalpy to the joules per kilogram per Kelvin form plotted in the graph below.

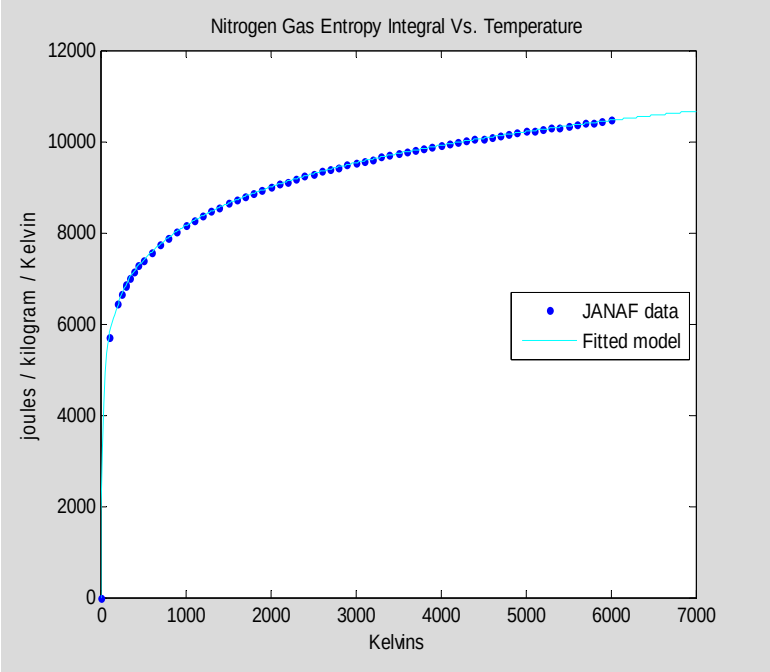
```
TK_S0_N2=[0;100;200;250;298.15;300;350;400;450;500;600;700;800;900;1000
;1100;1200;1300;1400;1500;1600;1700;1800;1900;2000;2100;2200;2300;2400;
2500;2600;2700;2800;2900;3000;3100;3200;3300;3400;3500;3600;3700;3800;3
900;4000;4100;4200;4300;4400;4500;4600;4700;4800;4900;5000;5100;5200;53
00;5400;5500;5600;5700;5800;5900;6000];
S0_N2=[0;159.811;179.985;186.481;191.609;191.789;196.281;200.181;
203.633;206.739;212.176;216.866;221.017;224.757;228.17;231.313;234.226;
236.943;239.487;241.88;244.138;246.275;248.304;250.234;252.074;253.833;
255.517;257.132;258.684;260.176;261.614;263.001;264.341;265.637;266.891
;
268.106;269.285;270.429;271.541;272.622;273.675;274.699;275.698;276.671
;
277.622;278.549;279.456;280.341;281.208;282.056;282.885;283.698;284.494
;
285.275;286.041;286.792;287.529;288.253;288.964;289.662;290.348;291.023
; 291.687;292.341;292.984];
plot(TK_S0_N2(2:end),S0_N2(2:end)/0.0280134,TK_S0_N2,S0_N2/0.0280134,'.
'), xlabel('Kelvins'),
ylabel('joules / kilogram / Kelvin'),
title('Nitrogen Gas Entropy Integral Vs. Temperature')
```



Next, we develop a leasst-squares spline fit to this data using a fixed knot sequence.

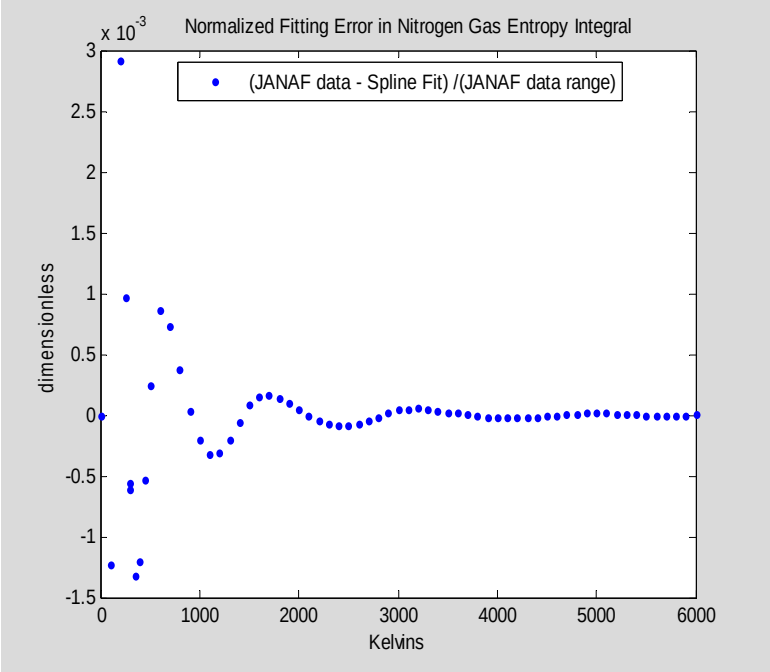
```
knots=augknt([0 100 500 1000 1500 2000 3000 4000 5000 6000],4);
S0_N2_pp=fn2fm(spap2(knots,4,TK_S0_N2,S0_N2/0.0280134),'pp');
S0_N2_fit=fnval(S0_N2_pp,linspace(0,7000,500));
plot(TK_S0_N2,S0_N2/0.0280134,'.',linspace(0,7000,500), S0_N2_fit,'-
c'), xlabel('Kelvins'),
ylabel('joules / kilogram / Kelvin'),
title('Nitrogen Gas Entropy Integral Vs. Temperature'),
legend('JANAF data','Fitted model','Location','East'),
format long g,
display(S0_N2_pp),
S0_N2_pp.coefs(:,1),
S0_N2_pp.coefs(:,2),
S0_N2_pp.coefs(:,3),
S0_N2_pp.coefs(:,4)

S0_N2_pp =
      form: 'pp'
    breaks: [0 100 500 1000 1500 2000 3000 4000 5000 6000]
     coefs: [9x4 double]
    pieces: 9
     order: 4
       dim: 1
ans =
       0.00476756426363939
     1.09803222755095e-005
     3.32401093977979e-007
     2.32031935274994e-007
     3.11648974413554e-008
     2.83500841216469e-008
     6.85986614938974e-009
     5.89420191844647e-009
      8.9804892608844e-010
ans =
         -1.44448598883726
       -0.0142167097454451
       -0.0010403230148337
     -0.000541721373866727
     -0.000193673470954222
      -0.00014692612479219
    -6.18758724272538e-005
    -4.12962739790874e-005
    -2.36136682237411e-005
ans =
          153.949611278738
          8.07934142046746
          1.97652831635594
          1.18550612200572
         0.817808699595229
         0.647508901722025
         0.438706904502589
         0.335534758096253
         0.270624815893414
ans =
```

```
                    0
         5717.66550314059
         7377.46913768897
         8147.20267890576
         8633.52938835131
         8997.91098259053
         9526.84384364201
         9910.53474186674
         10210.6674279024
```



Then we examine the residual discrepancies introduced by the spline fit.

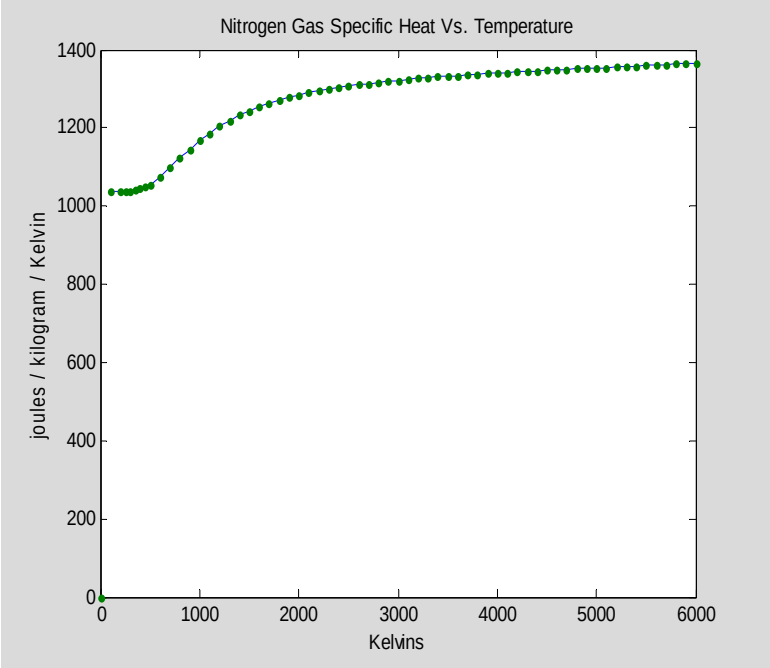
```
S0_N2_fit=fnval(S0_N2_pp,linspace(0,7000,500));
plot(TK_S0_N2,(S0_N2-0.0280134*fnval(S0_N2_pp,TK_S0_N2)) /(max(S0_N2)-
min(S0_N2)),'.'),
xlabel('Kelvins'),
ylabel('dimensionless'),
title('    Normalized Fitting Error in Nitrogen Gas Entropy Integral'),
legend('(JANAF data - Spline Fit) /(JANAF data range)','Location',
'North'),
```


Normalized Fitting Error in Nitrogen Gas Entropy Integral
x 10^-3
(JANAF data - Spline Fit) /(JANAF data range)
dimensionless
Kelvins
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
0
1000
2000
3000
4000
5000
6000

C.6.4 Nitrogen Specific Heat

Nitrogen gas specific heat at constant pressure (Cp) is taken from the published JANAF tables as follows. Specific heat values here are given in joules per mole per Kelvin and are tabulated vs temperature in Kelvins. The mass of a mole of nitrogen is 0.0280134 kilogram, a fact used to convert specific heat to the joules per kilogram per Kelvin form plotted in the graph below.trogen

```
TK_Cp_N2=[0;100;200;250;298.15;300;350;400;450;500;600;700;800;900;1000
;1100;1200;1300;1400;1500;1600;1700;1800;1900;2000;2100;2200;2300;2400;
2500;2600;2700;2800;2900;3000;3100;3200;3300;3400;3500;3600;3700;3800;3
900;4000;4100;4200;4300;4400;4500;4600;4700;4800;4900;5000;5100;5200;53
00;5400;5500;5600;5700;5800;5900;6000];
Cp_N2=[0;29.104;29.107;29.111;29.124;29.125;29.165;29.249;29.387;29.58;
30.11;30.754;31.433;32.09;32.697;33.241;33.723;34.147;34.518;34.843;
35.128;35.378;35.6;35.796;35.971;36.126;36.268;36.395;36.511;36.616;
36.713;36.801;36.883;36.959;37.03;37.096;37.158;37.216;37.271;37.323;
37.373;37.42;37.465;37.508;37.55;37.59;37.629;37.666;37.702;37.738;
37.773;37.808;37.843;37.878;37.912;37.947;37.981;38.013;38.046;38.08;
38.116;38.154;38.193; 38.234;38.276];
plot(TK_Cp_N2(2:end),Cp_N2(2:end)/0.0280134,TK_Cp_N2,Cp_N2/0.0280134,'.
'), xlabel('Kelvins'),
ylabel('joules / kilogram / Kelvin'),
title('Nitrogen Gas Specific Heat Vs. Temperature')
```

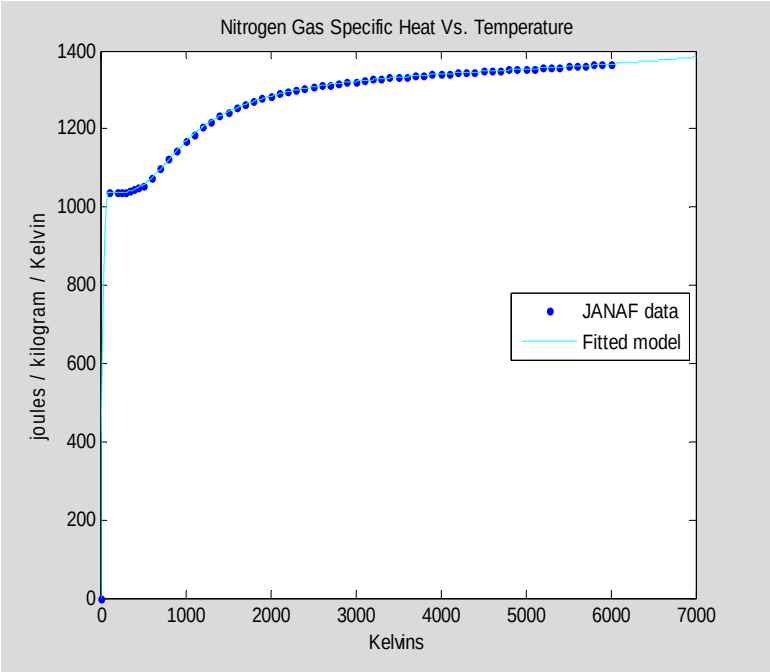


Next, we generate a spline approximation model fitted to the data.

```
knots=augknt([0 100 500 1000 1500 2000 3000 4000 5000 6000],4);
```

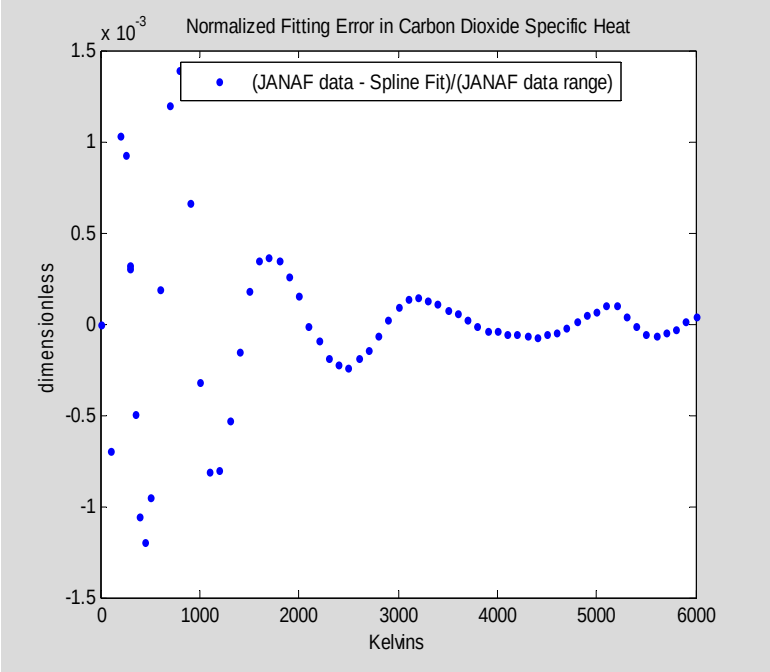
```
Cp_N2_pp=fn2fm(spap2(knots,4,TK_Cp_N2,Cp_N2/0.0280134),'pp');
Cp_N2_fit=fnval(Cp_N2_pp,linspace(000,7000,500));
plot(TK_Cp_N2,Cp_N2/0.0280134,'.',linspace(000,7000,500), Cp_N2_fit,'-
c'),
xlabel('Kelvins'),
ylabel('joules / kilogram / Kelvin'),
title('Nitrogen Gas Specific Heat Vs. Temperature'),
legend('JANAF data','Fitted model','Location','East'),
format long g,
display(Cp_N2_pp),
Cp_N2_pp.coefs(:,1),
Cp_N2_pp.coefs(:,2),
Cp_N2_pp.coefs(:,3),
Cp_N2_pp.coefs(:,4)

Cp_N2_pp =
      form: 'pp'
    breaks: [0 100 500 1000 1500 2000 3000 4000 5000 6000]
     coefs: [9x4 double]
    pieces: 9
     order: 4
       dim: 1
ans =
       0.00104538673122423
     1.25168670955324e-007
    -3.15200587312593e-007
     7.21128979254592e-008
      1.7626251638138e-008
     8.47558704019771e-009
     8.77635838424832e-010
     1.16135210370834e-009
     7.05012371759039e-010
ans =
        -0.313459220866882
      0.000156798500387563
      0.000307000905533953
     -0.000165799975434938
     -5.7630628546749e-005
    -3.11912510895422e-005
    -5.76448996894828e-006
     -3.1315824536739e-006
      3.5247385745197e-007
ans =
          31.2908448200112
       -0.0393974166383163
         0.146122345730289
         0.216722810779798
         0.105007508788954
        0.0605965689708082
        0.0236408279123164
        0.0147447554896942
        0.0119656468934709
ans =
                         0
          1039.87900455653
```

```
1057.21859290435
1167.62991873891
1243.55544251076
1283.85482122331
1321.73572614478
1340.48969992657
 1353.2642250663
```



Then we examine the approximation error introduced by the spline fit.

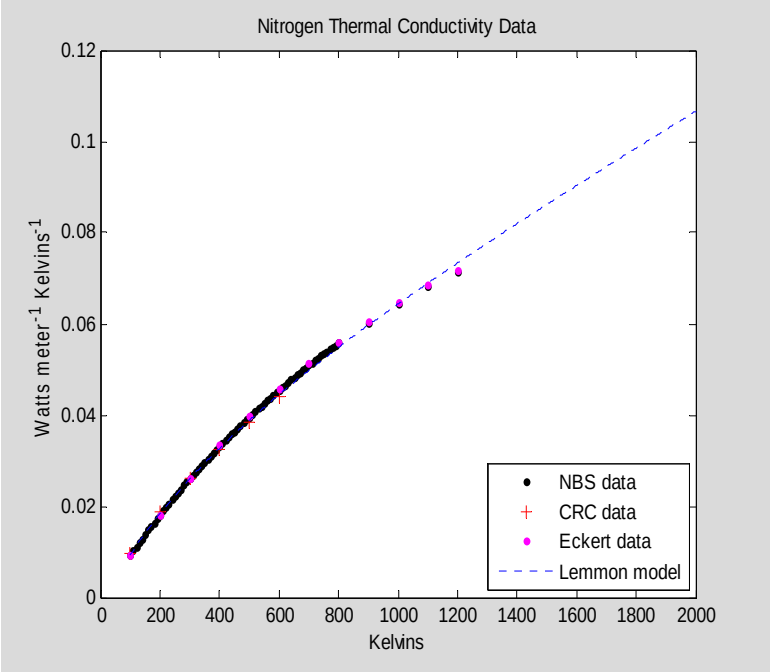
```
Cp_N2_fit=fnval(Cp_N2_pp,linspace(0,7000,500));
plot(TK_Cp_N2,(Cp_N2-0.0280134*fnval(Cp_N2_pp,TK_Cp_N2)) /(max(Cp_N2)-
min(Cp_N2)),'.'),
xlabel('Kelvins'),
ylabel('dimensionless'),
title('    Normalized Fitting Error in Carbon Dioxide Specific Heat'),
legend('(JANAF data - Spline Fit)/(JANAF data range)','Location',
'North'),
```


Normalized Fitting Error in Carbon Dioxide Specific Heat
x 10^-3
(JANAF data - Spline Fit)/(JANAF data range)
dimensionless
Kelvins
1.5
1
0.5
0
-0.5
-1
-1.5
0
1000
2000
3000
4000
5000
6000

C.6.5 Nitrogen Gas Thermal Conductivity

Tabulated thermal conductivity data for nitrogen gas was found from the CRC handbook, from NBS Circular 564, and in the book by Eckert. In addition, comprehensive model equations for nitrogen's thermal conductivity were published in a 2004 paper by Lemmon et al. Data from all of these sources have been entered here and are plotted below. Values are given in watts per meter per Kelvin, and are plotted versus temperature in Kelvins. Since the thermal conductivities of gases such as N2 are almost independent of pressure in the range of pressures relevant to this work, no pressure dependence is modeled here.

```
TK_N2ThrmCondNBS=[100;110;120;130;140;150;160;170;180;190;200;210;220;2
30;240;250;260;270;280;290;300;310;320;330;340;350;360;370;380;390;400;
410;420;430;440;450;460;470;480;490;500;510;520;530;540;550;560;570;580
;590;600;610;620;630;640;650;660;670;680;690;700;710;720;730;740;750;76
0;770;780;790;800;900;1000;1100;1200];
N2ThrmCondNBS=0.0241*
[0.390;0.427;0.465;0.502;0.538;0.576;0.612;0.648;0.684;0.719;0.753;0.78
9;0.823;0.857;0.892;0.924;0.957;0.990;1.021;1.051;1.081;1.111;1.141;1.1
72;1.202;1.232;1.262;1.292;1.321;1.349;1.377;1.405;1.433;1.460;1.487;1.
513;1.540;1.566;1.592;1.619;1.645;1.671;1.697;1.722;1.747;1.771;1.795;1
.819;1.843;1.867;1.890;1.913;1.936;1.959;1.982;2.005;2.027;2.048;2.070;
2.092;2.114;2.136;2.157;2.178;2.199;2.220;2.240;2.259;2.279;2.299;2.318
;2.504;2.673;2.828;2.968];
TK_N2ThrmCondEckert=[100;200;300;400;500;600;700;800;900;1000;1100;1200
];
N2ThrmCondEckert=[0.00945;0.01824;0.02620;0.03335;0.03984;0.04580;0.051
23;0.05609;0.06070;0.06475;0.06850;0.07184];
TK_N2ThrmCondCRC=[100;200;300;400;500;600];
N2ThrmCondCRC=1e-3*[9.8;18.7;26.0;32.3;38.3;44.0];

TK=(100:2000)';
ThrmCondN2Lemmon=(1.511E6*2.66958E-
8*sqrt(28.01348*TK)/(0.3656)^2./exp(0.431-
0.4623*log(TK/98.94)+0.08406*log(TK/98.94).^2+0.005341*log(TK/98.94).^3
-0.00331*log(TK/98.94).^4)+2.117*TK/150.687-
3.332*(TK/150.687).^0.7)/1000;
plot(TK_N2ThrmCondNBS,N2ThrmCondNBS,'.k',TK_N2ThrmCondCRC,N2ThrmCondCRC
,'+r',TK_N2ThrmCondEckert,N2ThrmCondEckert,'.m',TK,ThrmCondN2Lemmon,':b
'),
xlabel('Kelvins'),ylabel('Watts meter^-^1 Kelvins^-^1'),
title('Nitrogen Thermal Conductivity Data'),legend('NBS data','CRC
data','Eckert data','Lemmon model','Location','SouthEast')
```

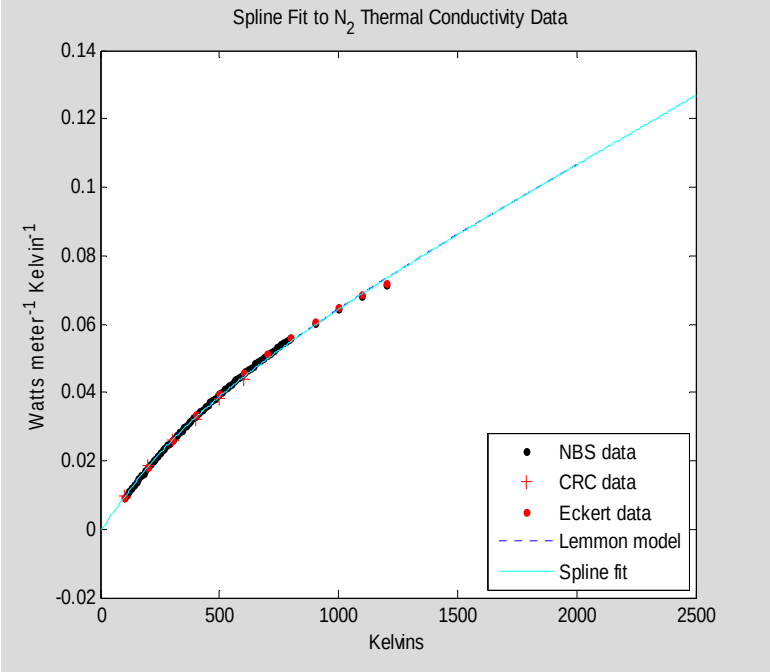


Next we approximate this data by using least squares techniques to fit cubic splines with a priori specified knot sequences.

```
knots=augknt([0 500 1000 1500 2000],4);
kth_N2_pp=fn2fm(spap2(knots,4,[TK_N2ThrmCondNBS; TK_N2ThrmCondCRC;
TK_N2ThrmCondEckert; TK], [N2ThrmCondNBS; N2ThrmCondCRC;
N2ThrmCondEckert; ThrmCondN2Lemmon]),'pp');
kth_N2_fit=fnval(kth_N2_pp,linspace(000,2500,500));
plot(TK_N2ThrmCondNBS,N2ThrmCondNBS,'.k',TK_N2ThrmCondCRC,
N2ThrmCondCRC,'+r', TK_N2ThrmCondEckert,N2ThrmCondEckert,'.r',
TK,ThrmCondN2Lemmon,':b',linspace(000,2500,500), kth_N2_fit,'-c'),
xlabel('Kelvins'),
ylabel('Watts meter^-^1 Kelvin^-^1'),
title('  Spline Fit to N_2 Thermal Conductivity Data '),
legend('NBS data','CRC data','Eckert data','Lemmon model','Spline
fit','Location','SouthEast')
format long g,
display(kth_N2_pp),
kth_N2_pp.coefs(:,1),
kth_N2_pp.coefs(:,2),
kth_N2_pp.coefs(:,3),
kth_N2_pp.coefs(:,4)
```

```
kth_N2_pp =
      form: 'pp'
    breaks: [0 500 1000 1500 2000]
     coefs: [4x4 double]
    pieces: 4
     order: 4
       dim: 1
```

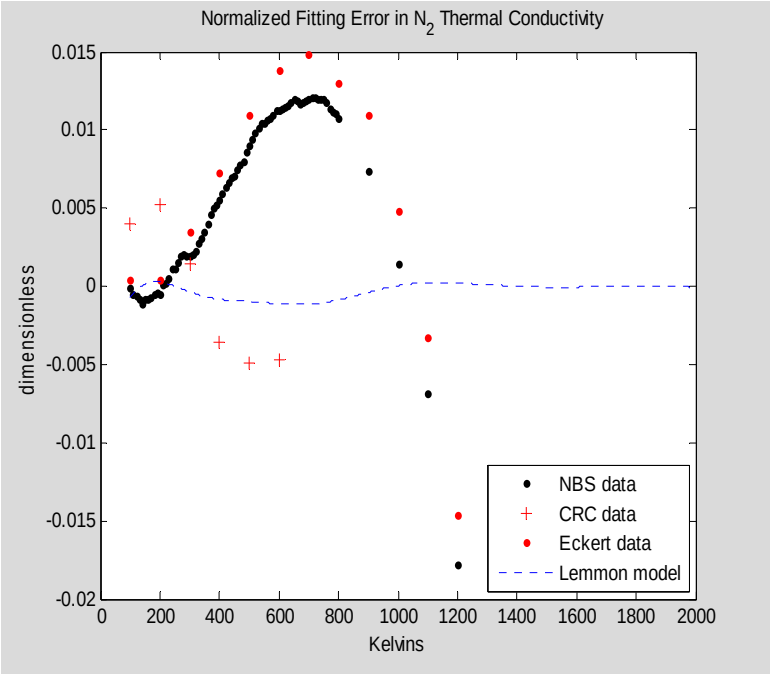
```
ans =
     3.90252304075642e-011
     1.05689751803369e-011
     1.38622644656266e-012
     1.82829825580045e-012
ans =
    -7.94217627419242e-008
    -2.08839171305778e-008
    -5.03045436007238e-009
    -2.95111469022847e-009
ans =
      0.000108965874681138
     5.88130347448874e-005
     4.58558489995622e-005
     4.18650644744119e-005
ans =
     -0.000732551043351641
        0.0387730994126821
        0.0642797594000234
        0.0861233486156067
```



Then we examine the approximation error introduced by the spline fit.

```
N2ThrmCondRange=max(ThrmCondN2Lemmon)-min(N2ThrmCondNBS);
plot(TK_N2ThrmCondNBS,(N2ThrmCondNBS-
fnval(kth_N2_pp,TK_N2ThrmCondNBS))/N2ThrmCondRange,'.k',
TK_N2ThrmCondCRC, (N2ThrmCondCRC-
fnval(kth_N2_pp,TK_N2ThrmCondCRC))/N2ThrmCondRange,'+r',
TK_N2ThrmCondEckert,(N2ThrmCondEckert-
fnval(kth_N2_pp,TK_N2ThrmCondEckert))/N2ThrmCondRange,'.r',
TK,(ThrmCondN2Lemmon-fnval(kth_N2_pp,TK))/N2ThrmCondRange,':b'),
xlabel('Kelvins'),
```

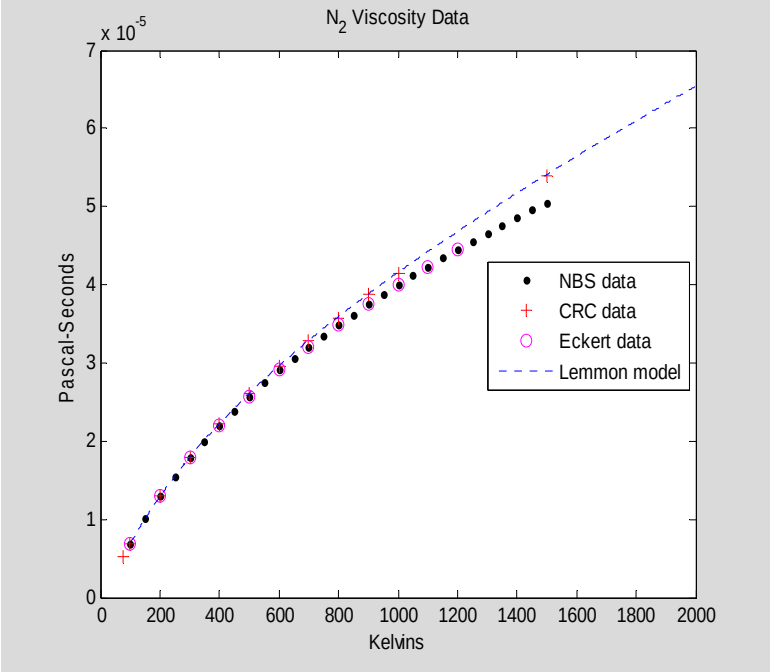
```
ylabel('dimensionless'),
title('  Normalized Fitting Error in N_2 Thermal Conductivity'),
legend('NBS data','CRC data','Eckert data','Lemmon
model','Location','SouthEast')
```



C.6.6 Nitrogen Gas Viscosity

Viscosity data for nitrogen gas was founed in the CRC handbook and in NBS Circular 564. In addition, a comprehensive model for nitrogen viscosity was published in a 2004 paper by Lemmon et al. Values here are given here in Pascal-seconds, and are tabulated versus temperature in Kelvins. Since the viscosities of gases such as nitrogen are almost independent of pressure in the range of pressures relevant to this work, no pressure dependence is modeled here.

```
TK_N2ViscEckert=[100;200;300;400;500;600;700;800;900;1000;1100;1200];
N2ViscEckert=1E-6*
[6.862;12.947;17.84;21.98;25.70;29.11;32.13;34.84;37.49;40.00;42.28;44.
50];
TK_N2ViscCRC=[77.25;100;200;300;400;500;600;700;800;900;1000;1500];   ,
N2ViscCRC=1e-
6*[5.3;6.8;12.9;18.0;22.2;26.1;29.5;32.8;35.8;38.7;41.5;54.0];
TK_N2ViscNBS=[100;150;200;250;300;350;400;450;500;550;600;650;700;750;8
00;850;899;950;1000;1050;1100;1150;1200;1250;1300;1350;1400;1450;1500];
N2ViscNBS=1.6625E-5*
[0.413;0.607;0.779;0.934;1.074;1.203;1.323;1.437;1.546;1.651;1.752;1.84
4;1.932;2.017;2.099;2.179;2.257;2.333;2.406;2.477;2.546;2.614;2.679;2.7
42;2.805;2.866;2.925;2.983;3.040];
TK=(100:2000)';
ViscN2Lemmon=2.66958E-8*sqrt(28.01348*TK)/(0.3656)^2./exp(0.431-
0.4623*log(TK/98.94)+0.08406*log(TK/98.94).^2+0.005341*log(TK/98.94).^3
-0.00331*log(TK/98.94).^4);

plot(TK_N2ViscNBS,N2ViscNBS,'.k',
TK_N2ViscCRC,N2ViscCRC,'+r',TK_N2ViscEckert,N2ViscEckert,'om',TK,ViscN2
Lemmon,':b'),title('N_2 Viscosity Data'),
xlabel('Kelvins'),
ylabel('Pascal-Seconds'),
legend('NBS data','CRC data','Eckert data','Lemmon
model','Location','East')
```

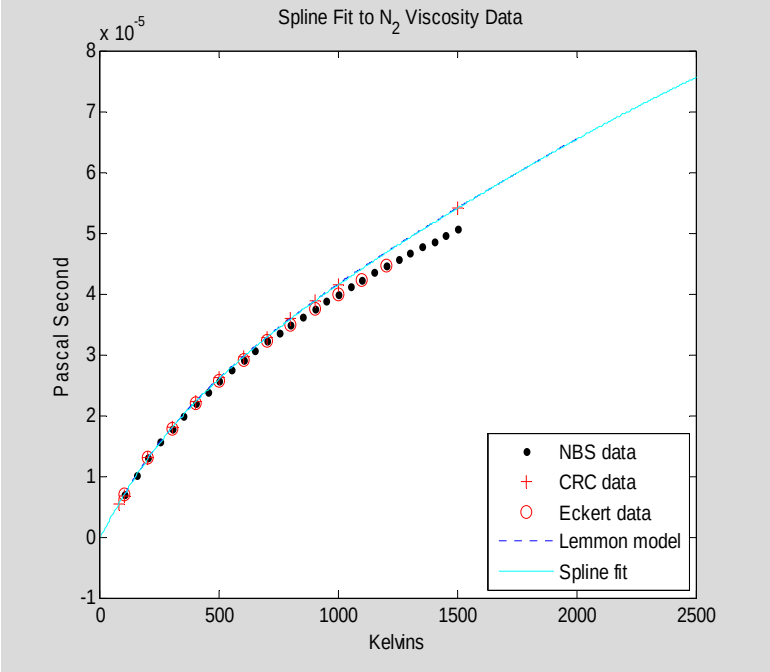


Next we approximate this data by using least squares techniques to fit a cubic spline with knot sequences specified a priori.

```
knots=augknt([0 500 1000 1500 2000],4);
mu_N2_pp=fn2fm(spap2(knots,4,[TK_N2ViscNBS; TK_N2ViscCRC;
TK_N2ViscEckert; TK], [N2ViscNBS; N2ViscCRC; N2ViscEckert;
ViscN2Lemmon]),'pp');
mu_N2_fit=fnval(mu_N2_pp,linspace(000,2500,500));
plot(TK_N2ViscNBS,N2ViscNBS,'.k',TK_N2ViscCRC,N2ViscCRC,'+r',
TK_N2ViscEckert,N2ViscEckert,'or', TK,ViscN2Lemmon,':b',
linspace(000,2500,500), mu_N2_fit,'-c'),
xlabel('Kelvins'),
ylabel('Pascal Second'),
title('  Spline Fit to N_2 Viscosity Data '),
legend('NBS data','CRC data','Eckert data','Lemmon model','Spline
fit','Location','SouthEast')
format long g,
display(mu_N2_pp),
mu_N2_pp.coefs(:,1),
mu_N2_pp.coefs(:,2),
mu_N2_pp.coefs(:,3),
mu_N2_pp.coefs(:,4)
```

```
mu_N2_pp =
      form: 'pp'
    breaks: [0 500 1000 1500 2000]
     coefs: [4x4 double]
    pieces: 4
     order: 4
       dim: 1
```

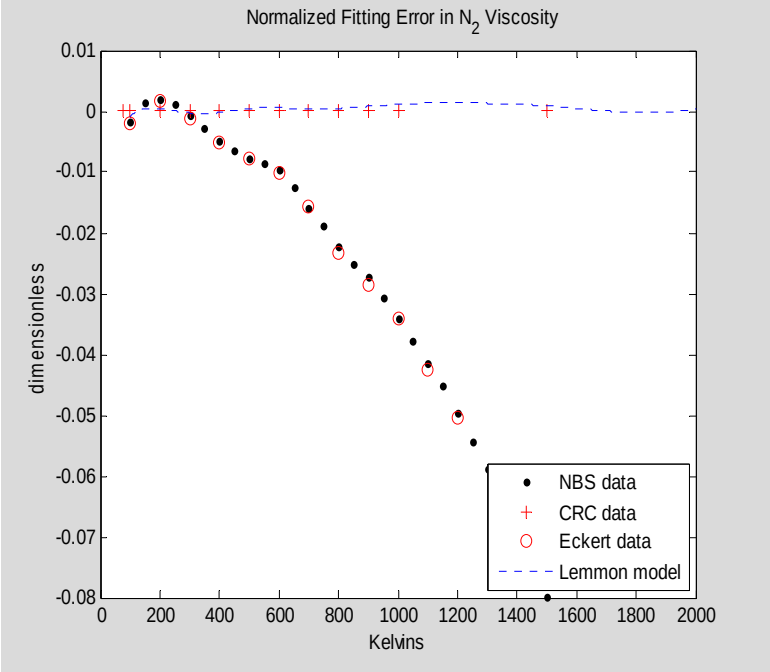
```
ans =
     3.43204234160597e-014
     6.45842982731327e-015
     1.93145417908812e-015
     -3.5605237744431e-016
ans =
    -6.58717146857726e-011
    -1.43910795616831e-011
    -4.70343482071328e-012
    -1.80625355208116e-012
ans =
     7.66182495560525e-008
     3.64868524323247e-008
     2.69395952411266e-008
     2.36847510547294e-008
ans =
    -9.49371077254447e-008
     2.60363119258651e-005
     4.14892719800209e-005
     5.40246426677918e-005
```



Next we examine the approximation error introduced by this spline fit.

```
N2ViscRange=max(N2ViscNBS)-min(N2ViscNBS);
plot(TK_N2ViscNBS,(N2ViscNBS-
fnval(mu_N2_pp,TK_N2ViscNBS))/N2ViscRange,'.k',
TK_N2ViscCRC,N2ViscCRC,'+r',TK_N2ViscEckert,(N2ViscEckert-
fnval(mu_N2_pp,TK_N2ViscEckert))/N2ViscRange,'or',
TK,(ViscN2Lemmon-fnval(mu_N2_pp,TK))/N2ViscRange,':b'),
xlabel('Kelvins'),
ylabel('dimensionless'),
title('  Normalized Fitting Error in N_2 Viscosity'),
```

```
legend('NBS data','CRC data','Eckert data','Lemmon model',
'Location','SouthEast')
```



C.7 MarsMix Gas Properties

C.7.1 MarsMix Gas Density

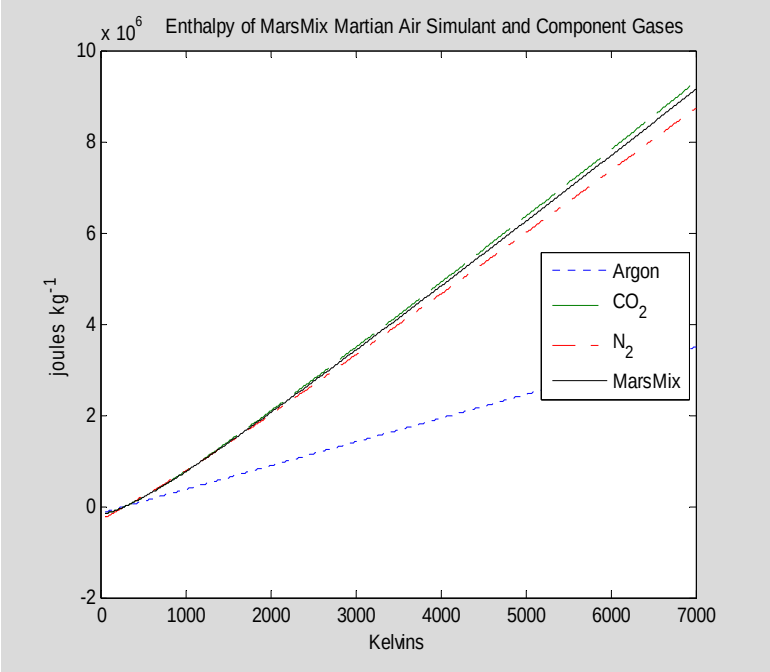
$$\rho = \frac{p}{RT} \text{ where } R = \frac{\mathcal{R}}{\hat{M}} = \frac{8.31361\,\text{Joule/mole/Kelvin}}{0.043513\,\text{kg/mole}} = 191.06037\,\text{Joule kg}^{-1}\,\text{Kelvin}^{-1}$$

where $0.043513\,\text{kg/mole}$ is the molal mass of MarsMix gas .

C.7.2 MarsMix Gas Enthalpy

```
H0_MarsMix_pp=H0_CO2_pp;
H0_MarsMix_pp.coefs=0.957*H0_CO2_pp.coefs+0.027*H0_Ar_pp.coefs+0.016*H0
_N2_pp.coefs;
T_plot=linspace(50,7000,1000);
H0_MarsMix_plot=fnval(H0_MarsMix_pp,T_plot);
H0_Ar_plot=fnval(H0_Ar_pp,T_plot);
H0_CO2_plot=fnval(H0_CO2_pp,T_plot);
H0_N2_plot=fnval(H0_N2_pp,T_plot);
plot(T_plot,H0_Ar_plot,':',T_plot,H0_CO2_plot,'--',T_plot,H0_N2_plot,'-
.',T_plot,H0_MarsMix_plot,'-k'),
xlabel('Kelvins'),
ylabel('joules kg^-^1'),
title('          Enthalpy of MarsMix Martian Air Simulant and Component
Gases'),
legend('Argon','CO_2','N_2','MarsMix','Location','East'),
H0_MarsMix_pp.form,
H0_MarsMix_pp.breaks',
H0_MarsMix_pp.pieces,
H0_MarsMix_pp.order,
H0_MarsMix_pp.dim,
H0_MarsMix_pp.coefs(:,1),
H0_MarsMix_pp.coefs(:,2),
H0_MarsMix_pp.coefs(:,3),
H0_MarsMix_pp.coefs(:,4)

ans =
pp
ans =
           0
         100
         500
        1000
        1500
        2000
        3000
        4000
        5000
        6000
```

```
ans =
     9
ans =
     4
ans =
     1
ans =
        0.0102274805269917
     -0.000261293627148073
     -0.000127890163699401
     -4.0491846917942e-005
    -2.00615876268502e-005
    -5.13909684719186e-006
    -1.20579512292617e-006
    -3.54058916457128e-007
     1.31268140418166e-006
ans =
         -2.44358425545878
          0.624659902638749
          0.311107550061062
          0.119272304511961
         0.0585345341350491
         0.0284421526947733
         0.0130248621531977
        0.00940747678442164
        0.00834530003504754
ans =
          806.998446424539
          625.106011142536
           999.41299222246
          1214.60291950897
          1303.50633883248
          1346.99468224739
          1388.46169709536
          1410.89403603297
          1428.64681285245
ans =
         -212761.983720458
           -146270.5011056
          186994.695636138
          748491.808800208
          1380549.86381794
          2044428.96831459
          3414726.70640956
          4815007.47053519
          6234954.92443613
```

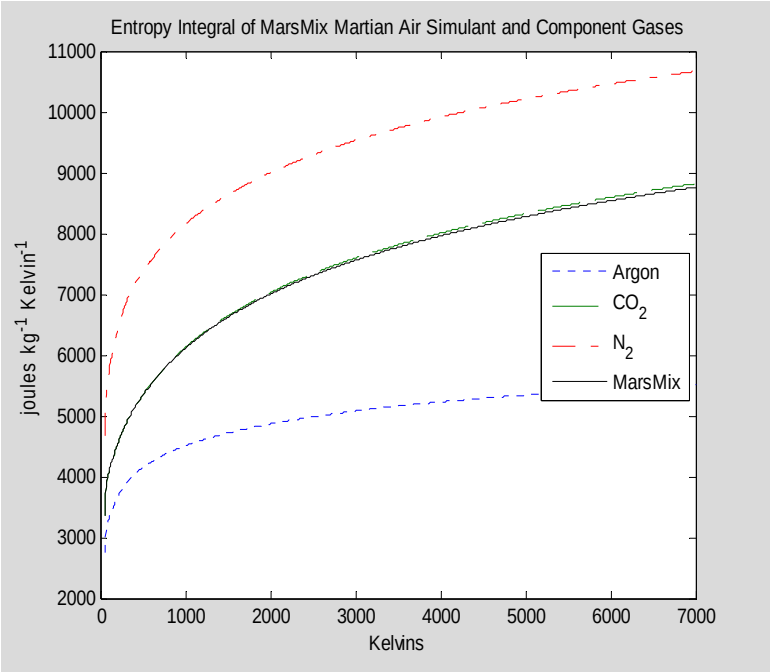


C.7.3 MarsMix Gas Entropy Integral

```
S0_MarsMix_pp=S0_CO2_pp;
S0_MarsMix_pp.coefs=0.957*S0_CO2_pp.coefs+0.027*S0_Ar_pp.coefs+0.016*S0
_N2_pp.coefs;
T_plot=linspace(50,7000,1000);
S0_MarsMix_plot=fnval(S0_MarsMix_pp,T_plot);
S0_Ar_plot=fnval(S0_Ar_pp,T_plot);
S0_CO2_plot=fnval(S0_CO2_pp,T_plot);
S0_N2_plot=fnval(S0_N2_pp,T_plot);
plot(T_plot,S0_Ar_plot,':',T_plot,S0_CO2_plot,'--',T_plot,S0_N2_plot,'-
.',T_plot,S0_MarsMix_plot,'-k'),
xlabel('Kelvins'),
ylabel('joules kg^-^1 Kelvin^-^1'),
title('Entropy Integral of MarsMix Martian Air Simulant and Component
Gases'),
legend('Argon','CO_2','N_2','MarsMix','Location','East'),

S0_MarsMix_pp.form,
S0_MarsMix_pp.breaks',
S0_MarsMix_pp.pieces,
S0_MarsMix_pp.order,
S0_MarsMix_pp.dim,
S0_MarsMix_pp.coefs(:,1),
S0_MarsMix_pp.coefs(:,2),
S0_MarsMix_pp.coefs(:,3),
S0_MarsMix_pp.coefs(:,4)
```

```
ans =
pp
ans =
           0
         100
         500
        1000
        1500
        2000
        3000
        4000
        5000
        6000
ans =
     9
ans =
     4
ans =
     1
ans =
         0.003485573924265
     5.29285604313321e-006
     2.86919972067194e-007
      1.7787566394139e-007
     5.21139189360539e-008
     2.71444966661034e-008
     8.21911815563496e-009
      5.5373104734041e-009
     1.83880168983344e-009
ans =
          -1.0529469881161
       -0.0072748108366035
     -0.000923383584843663
     -0.000493003626742864
     -0.000226190130830779
     -0.000148019252426702
    -6.65857624283913e-005
    -4.19284079614855e-005
    -2.53164765412724e-005
ans =
          111.232409185315
          5.21022929004383
          1.93095152146498
          1.22275791567171
         0.863161036884889
          0.67605634525615
         0.461451330401057
         0.352937160011179
         0.285692275508419
ans =
                         0
          4079.34496163542
          5338.20973055692
          6108.70459158689
           6619.0671007297
```

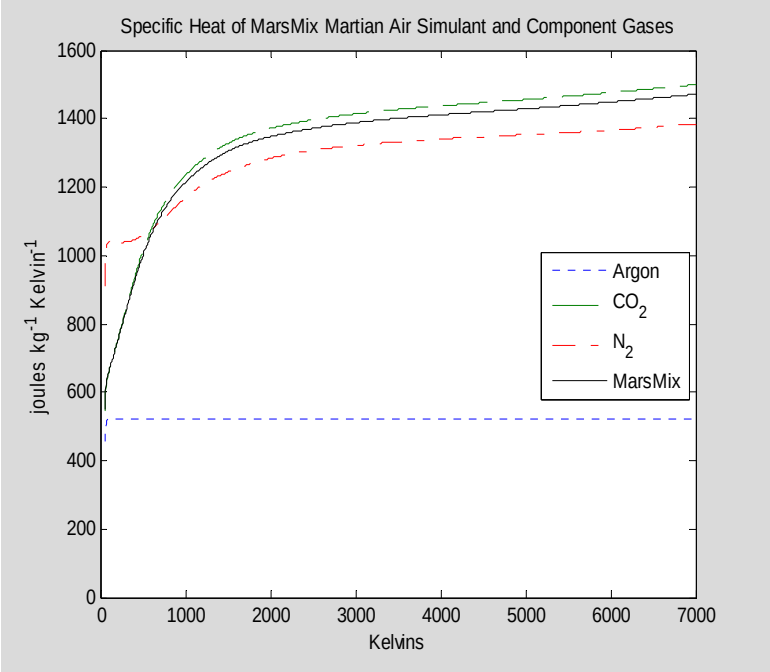
```
7000.61432633146
7555.79591582701
7958.88060195531
8275.42666447841
```



C.7.4 MarsMix Gas Specific Heat

```
Cp_MarsMix_pp=Cp_CO2_pp;
Cp_MarsMix_pp.coefs=0.957*Cp_CO2_pp.coefs+0.027*Cp_Ar_pp.coefs+0.016*Cp
_N2_pp.coefs;
Cp_MarsMix_pp.coefs(:,1),
Cp_MarsMix_pp.coefs(:,2),
Cp_MarsMix_pp.coefs(:,3),
Cp_MarsMix_pp.coefs(:,4),
T_plot=linspace(50,7000,1000);
Cp_MarsMix_plot=fnval(Cp_MarsMix_pp,T_plot);
Cp_CO2_plot=fnval(Cp_CO2_pp,T_plot);
Cp_N2_plot=fnval(Cp_N2_pp,T_plot);
Cp_Ar_plot=fnval(Cp_Ar_pp,T_plot);
plot(T_plot,Cp_Ar_plot,':',T_plot,Cp_CO2_plot,'--',T_plot,Cp_N2_plot,'-
.',T_plot,Cp_MarsMix_plot,'-k'),
ylim([0 1600])
xlabel('Kelvins'),
ylabel('joules kg^-^1 Kelvin^-^1'),
```

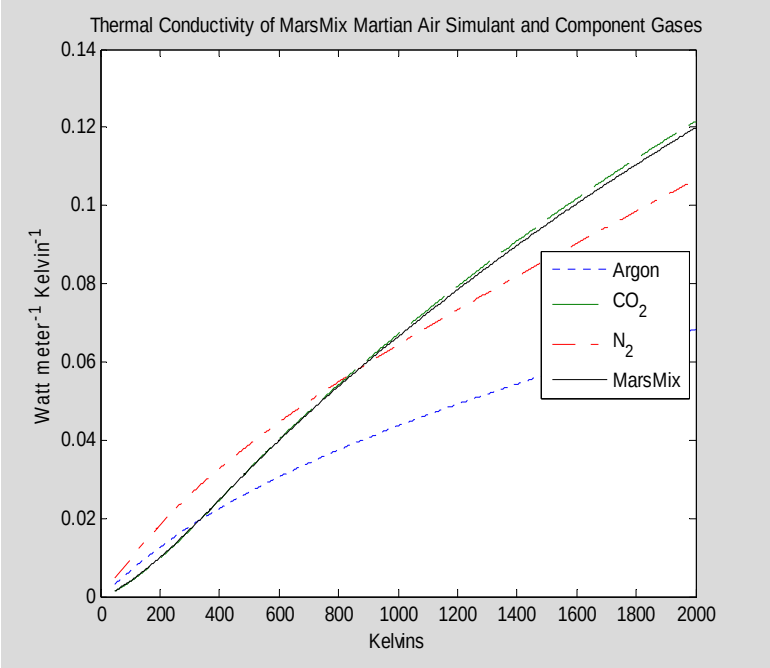
```
title('Specific Heat of MarsMix Martian Air Simulant and Component
Gases'),
legend('Argon','CO_2','N_2','MarsMix','Location','East')

ans =
      0.000581350983630857
    -9.65450015550039e-007
     3.98026528343767e-007
      3.3614634225973e-008
     5.11730668098727e-008
     5.88396970721698e-009
     1.61139384960509e-009
      1.4224809375556e-009
    -1.59296836722557e-010
ans =
        -0.173995861194196
      0.000409433895060723
     -0.000749106123599324
     -0.000152066331083674
     -0.000101644379744716
    -2.48847795299067e-005
    -7.23287040825545e-006
    -2.39868885944164e-006
     1.86875395322404e-006
ans =
          18.1956493279625
         0.837006598048948
         0.701137706633507
         0.250551479292009
         0.123696123877816
        0.0604315442405037
        0.0283138943023409
         0.018682335034646
        0.0181524001284304
ans =
                         0
          660.957304485144
          999.480565919237
          1212.52620437913
          1303.98719053246
          1346.82079088643
          1388.25152530424
          1410.94394304793
          1428.65007016069
```



C.7.5 MarsMix Gas Thermal Conductivity

```
kth_MarsMix_pp=kth_CO2_pp;
kth_MarsMix_pp.coefs=0.957*kth_CO2_pp.coefs+0.027*kth_Ar_pp.coefs+0.016
*kth_N2_pp.coefs;
kth_MarsMix_pp
kth_MarsMix_pp.breaks
kth_MarsMix_pp.coefs(:,1),
kth_MarsMix_pp.coefs(:,2),
kth_MarsMix_pp.coefs(:,3),
kth_MarsMix_pp.coefs(:,4),
T_plot1=linspace(50,2000,1000);
kth_MarsMix_plot=fnval(kth_MarsMix_pp,T_plot1);
kth_Ar_plot=fnval(kth_Ar_pp,T_plot1);
kth_CO2_plot=fnval(kth_CO2_pp,T_plot1);
kth_N2_plot=fnval(kth_N2_pp,T_plot1);
plot(T_plot1,kth_Ar_plot,':',T_plot1,kth_CO2_plot,'--',
T_plot1,kth_N2_plot,'-.',T_plot1,kth_MarsMix_plot,'-k'),
xlabel('Kelvins'),
ylabel('Watt meter^-^1 Kelvin^-^1'),
title('Thermal Conductivity of MarsMix Martian Air Simulant and
Component Gases'),
legend('Argon','CO_2','N_2','MarsMix','Location','East')

kth_MarsMix_pp =
      form: 'pp'
    breaks: [0 500 1000 1500 2000]
     coefs: [4x4 double]
    pieces: 4
```

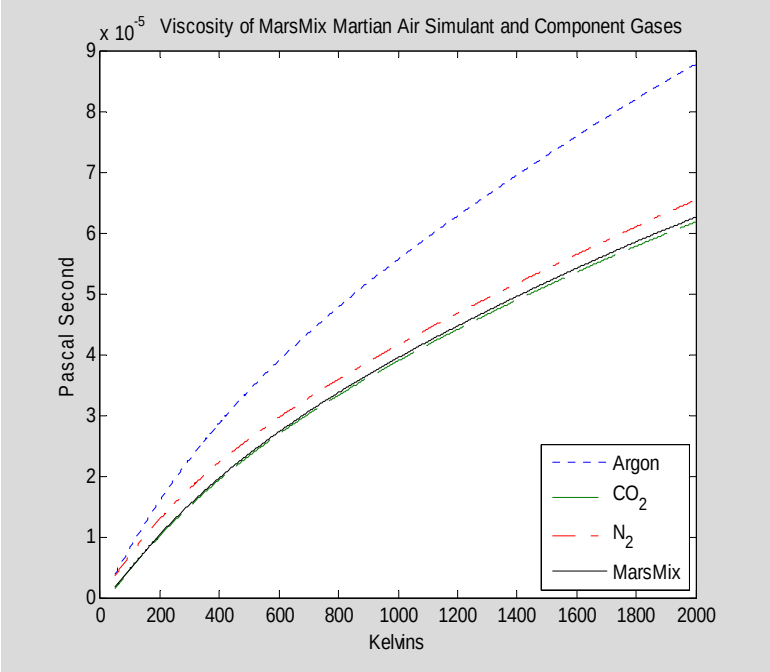
```
     order: 4
       dim: 1
ans =
           0         500        1000        1500        2000
ans =
    -8.43946965540552e-011
     8.68829314643623e-012
    -1.63399992488335e-013
     1.20614460800434e-012
ans =
     1.05356931304181e-007
    -2.12351135269022e-008
    -8.20267380724793e-009
    -8.44777379598044e-009
ans =
     3.43346633193316e-005
     7.63955722079707e-005
     6.16766785408957e-005
     5.33514547392816e-005
ans =
    -0.000563169020844222
       0.0323940583956098
       0.0663691027611741
       0.0951363485807489
```



C.7.6 MarsMix Gas Viscosity

```
mu_MarsMix_pp=mu_CO2_pp;
```

```
mu_MarsMix_pp.coefs=0.957*mu_CO2_pp.coefs+0.027*mu_Ar_pp.coefs+0.016*mu
_N2_pp.coefs;
mu_MarsMix_pp.coefs(:,1),
mu_MarsMix_pp.coefs(:,2),
mu_MarsMix_pp.coefs(:,3),
mu_MarsMix_pp.coefs(:,4),
T_plot1=linspace(50,2000,1000);
mu_MarsMix_plot=fnval(mu_MarsMix_pp,T_plot1);
mu_Ar_plot=fnval(mu_Ar_pp,T_plot1);
mu_CO2_plot=fnval(mu_CO2_pp,T_plot1);
mu_N2_plot=fnval(mu_N2_pp,T_plot1);
plot(T_plot1,mu_Ar_plot,':',T_plot1,mu_CO2_plot,'--',
T_plot1,mu_N2_plot,'-.',T_plot1,mu_MarsMix_plot,'-k'),
xlabel('Kelvins'),
ylabel('Pascal Second'),
title('         Viscosity of MarsMix Martian Air Simulant and Component
Gases'),
legend('Argon','CO_2','N_2','MarsMix','Location','SouthEast')

ans =
     1.91259324527568e-014
     7.06525325429854e-015
     1.59489983582924e-015
    -1.36251102956505e-015
ans =
     -4.4580839763486e-011
    -1.58919410843509e-011
    -5.29406120290307e-012
    -2.90171144915931e-012
ans =
     6.80145875596715e-008
     3.77781971357531e-008
      2.7185195992126e-008
     2.30873096660949e-008
ans =
    -1.68253261087149e-006
     2.35702927846874e-005
     3.93695627382635e-005
     5.18380079130794e-005
```



C.8 EarthAir Gas Properties

Two recent publications by Eric W. Lemmon et al, and the much older 1955 NBS Circular 564, are chosen as sources for thermodynamic properties of EarthAir. The low pressure "perfect gas" limits of the data are herein fitted with cubic splines to facilitate their use.

C.8.1 EarthAir Gas Density

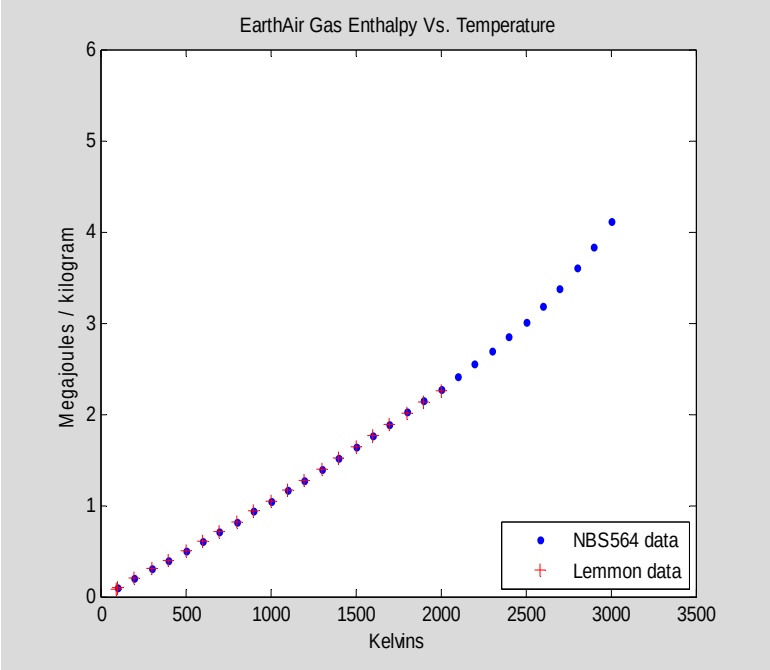
$$\rho = \frac{p}{RT} \text{ where } R = \frac{\mathcal{R}}{\hat{M}} = \frac{8.31361 \text{ Joule/mole/Kelvin}}{0.028966 \text{ kg/mole}} = 287.0127 \text{ Joule kg}^{-1} \text{ Kelvin}^{-1}$$

where 0.028966 kg/mole is the molal mass of MarsMix gas .

C.8.2 EarthAir Gas Enthalpy

EarthAir enthalpy values are selected from entries in the 1 atm isobar of NBS Circular 564 tables for air. Enthalpy values here, in joules per mole, are tabulated vs temperatures in Kelvins. The mass of a mole of argon is 0.028966 kilogram, a fact used to convert enthalpy to the megajoules per kilogram form plotted in the graph below.

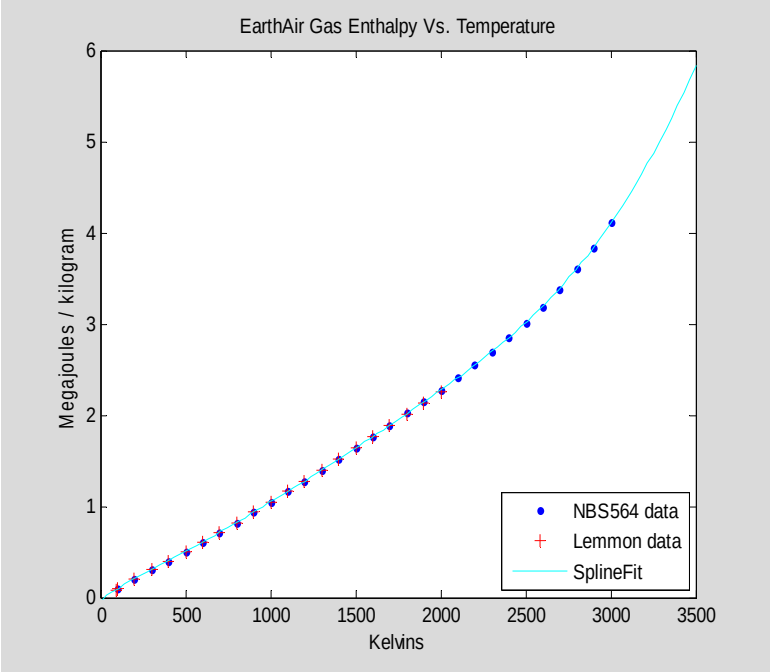
```
TK_H0_EarthAirNBS=[100; 200; 300; 400; 500; 600; 700; 800; 900; 1000;
1100; 1200; 1300; 1400; 1500; 1600; 1700; 1800; 1900; 2000; 2100; 2200;
2300; 2400; 2500; 2600; 2700; 2800; 2900; 3000];
H0_EarthAirNBS=78407.9*[1.2552; 2.5465; 3.8292; 5.1167; 6.4195; 7.7463;
9.1023; 10.489; 11.904; 13.348; 14.817; 16.310; 17.826; 19.363; 20.922;
22.504; 24.110; 25.740; 27.397; 29.086; 30.813; 32.592; 34.443; 36.393;
38.470; 40.713; 43.172; 45.901; 48.960; 52.403];
TK_H0_EarthAirLemmon=[90; 100; 200; 300; 400; 500; 600; 700; 800; 900;
1000; 1100; 1200; 1300; 1400; 1500; 1600; 1700; 1800; 1900; 2000];
H0_EarthAirLemmon=[2538.2; 2842.1; 5782.9; 8696.5; 11621.0; 14579.0;
17592.0; 20671.0; 23820.; 27035.;30311; 33642; 37022.; 40444; 43904;
47397; 50919; 54466; 58038; 61630; 65242];
plot(TK_H0_EarthAirNBS,H0_EarthAirNBS/1e6,'.b', TK_H0_EarthAirLemmon,
H0_EarthAirLemmon/0.028966/1e6,'+r'),
ylim([0 6]),
xlim([0 3500])
xlabel('Kelvins'),
ylabel('Megajoules / kilogram'),
title('EarthAir Gas Enthalpy Vs. Temperature'),
legend('NBS564 data','Lemmon data','Location','SouthEast')
```



Next we approximate this data by using least squares techniques to fit cubic splines with a priori specified knot sequences.

```
knots=augknt([0 100 500 1000 1500 2000 3000],4);
H0_EarthAir_pp=fn2fm(spap2(knots,4,
[TK_H0_EarthAirNBS;TK_H0_EarthAirLemmon],
[H0_EarthAirNBS;H0_EarthAirLemmon/0.028966]),'pp');
H0_EarthAir_fit=fnval(H0_EarthAir_pp,linspace(000,3500));
```

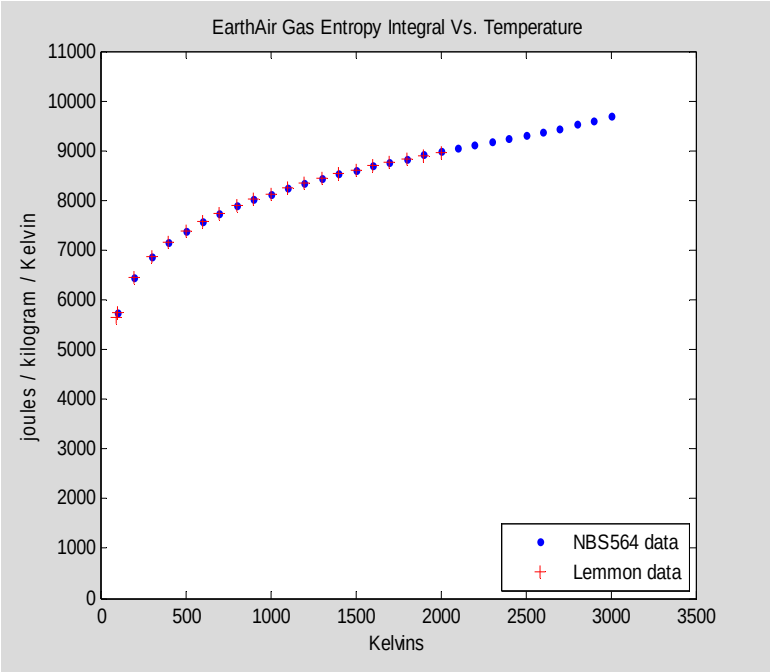
```
plot(TK_H0_EarthAirNBS,H0_EarthAirNBS/1e6,'.b', TK_H0_EarthAirLemmon,
H0_EarthAirLemmon/0.028966/1e6,'+r',
linspace(000,3500),H0_EarthAir_fit/1e6,'-c'),
ylim([0 6]);
xlabel('Kelvins'),
ylabel('Megajoules / kilogram'),
title('EarthAir Gas Enthalpy Vs. Temperature'),
legend('NBS564 data','Lemmon data','SplineFit','Location','SouthEast')
format long g,
display(H0_EarthAir_pp),
H0_EarthAir_pp.coefs(:,1),
H0_EarthAir_pp.coefs(:,2),
H0_EarthAir_pp.coefs(:,3),
H0_EarthAir_pp.coefs(:,4)

H0_EarthAir_pp =
      form: 'pp'
    breaks: [0 100 500 1000 1500 2000 3000]
     coefs: [6x4 double]
    pieces: 6
     order: 4
       dim: 1
ans =
         0.027864725845585
       0.00025695302835123
    -4.94037917273004e-005
     1.73550880724561e-005
     7.74163690522254e-006
      0.000389207628736484
ans =
        -8.52127015766061
        -0.161852403985123
         0.146491230036354
        0.0723855424454055
        0.0984181745540859
         0.110030629911925
ans =
          1903.23433174482
          1034.92207558025
          1028.77760600074
          1138.21599224162
          1223.61785074137
          1327.84225297437
ans =
         -34955.3138359624
          98020.1436074985
          502537.583016457
             1047373.71956
          1636747.48730122
          2274128.66092358
```



C.8.3 EarthAir Gas Entropy Integral

EarthAirentropy integral values are taken from the NBS564 tables for air, and from a recent publication by Lemmon et al. Entropy integral values here are given in joules per mole per Kelvin and are tabulated vs temperature in Kelvins. The mass of a mole of EarthAir is 0.028966 kilogram, a fact used to convert enthalpy to the joules per kilogram per Kelvin form plotted in the graph below.

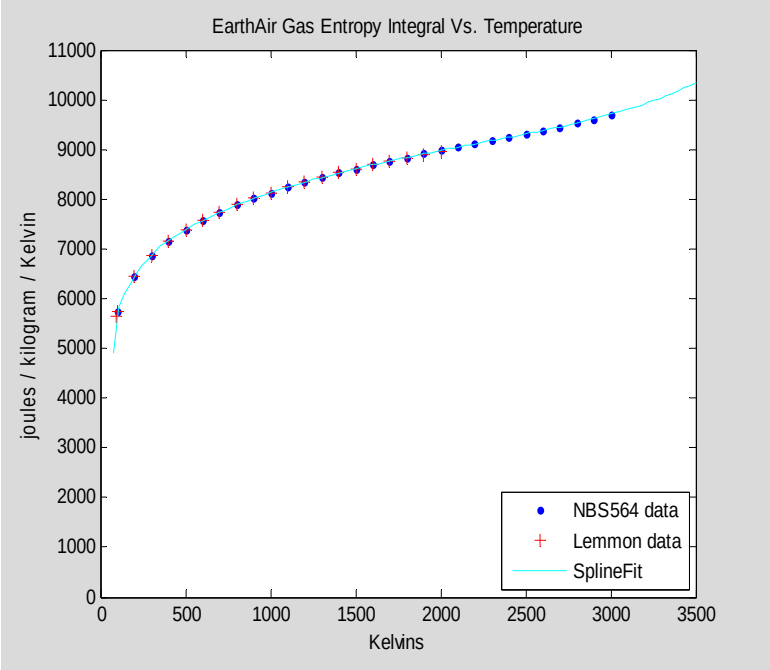
```
TK_S0_EarthAirNBS=[100;200;300;400;500;600;700;800;900;1000;1100;1200;1
300;1400;1500;1600;1700;1800;1900;2000;2100;2200;2300;2400;
2500;2600;2700;2800;2900;3000];
S0_EarthAirNBS=287.041*[20.049; 22.497; 23.917; 24.929; 25.723; 26.383;
26.954; 27.460; 27.915; 28.330; 28.713; 29.068; 29.399; 29.711; 30.005;
30.284; 30.549; 30.804; 31.048; 31.284; 31.514; 31.740; 31.964; 32.191;
32.423; 32.663; 32.917; 33.188; 33.481; 33.799];
TK_S0_EarthAirLemmon=[90;100;200;300;400;500;600;700;800;900;1000;1100;
1200;1300;1400;1500;1600;1700;1800;1900;2000];
S0_EarthAirLemmon=[163.41; 166.61; 187.03; 198.85; 207.26; 213.86;
219.35; 224.09; 228.30; 232.08; 235.53; 238.71; 241.65; 244.39; 246.95;
249.36; 251.63; 253.79; 255.83; 257.77; 259.62]/0.028966;
plot(TK_S0_EarthAirNBS,S0_EarthAirNBS,'.b',TK_S0_EarthAirLemmon,S0_Eart
hAirLemmon,'+r'),
xlim([0 3500])
ylim([0 11000])
xlabel('Kelvins'),
ylabel('joules / kilogram / Kelvin'),
title('EarthAir Gas Entropy Integral Vs. Temperature'),
legend('NBS564 data','Lemmon data','Location','SouthEast')
```



Next, we develop a least-squares spline fit to this data using a fixed knot sequence.

```
knots=augknt([0 100 500 1000 1500 2000 2500 3000 ],4);
S0_EarthAir_pp=fn2fm(spap2(knots,4,[TK_S0_EarthAirNBS;TK_S0_EarthAirLem
mon],[S0_EarthAirNBS;S0_EarthAirLemmon]),'pp');
S0_EarthAir_pp.coefs(:,1),
S0_EarthAir_pp.coefs(:,2),
S0_EarthAir_pp.coefs(:,3),
S0_EarthAir_pp.coefs(:,4),
S0_EarthAir_fit=fnval(S0_EarthAir_pp,linspace(75,3500) );
plot(TK_S0_EarthAirNBS,S0_EarthAirNBS,'.b',TK_S0_EarthAirLemmon,S0_Eart
hAirLemmon,'+r',linspace(75,3500), S0_EarthAir_fit,'-c'),
xlim([0 3500])
ylim([0 11000])
xlabel('Kelvins'),
ylabel('joules / kilogram / Kelvin'),
title('EarthAir Gas Entropy Integral Vs. Temperature'),
legend('NBS564 data','Lemmon data','SplineFit','Location','SouthEast')

ans =
        0.0414394391355335
     1.09913650898902e-005
     2.79250504586173e-007
     2.47285910454356e-007
     5.74355887654054e-008
     1.13455667339807e-007
     2.70403090412234e-007
ans =
         -12.4459809061517
```

```
       -0.0141491654916217
     -0.000959527383753384
     -0.000540651626874114
     -0.000169722761192574
    -8.35693780444491e-005
     8.66141229652566e-005
ans =
          1253.96538745124
          7.95238028691227
          1.90890313676224
          1.15881363144848
         0.803626437415129
         0.676980367796601
         0.678502740257016
ans =
         -36612.3489703295
          5763.81984881136
          7384.35285066977
          8133.82888618582
          8608.98353399832
          8975.54551100342
          9307.32530880808
```



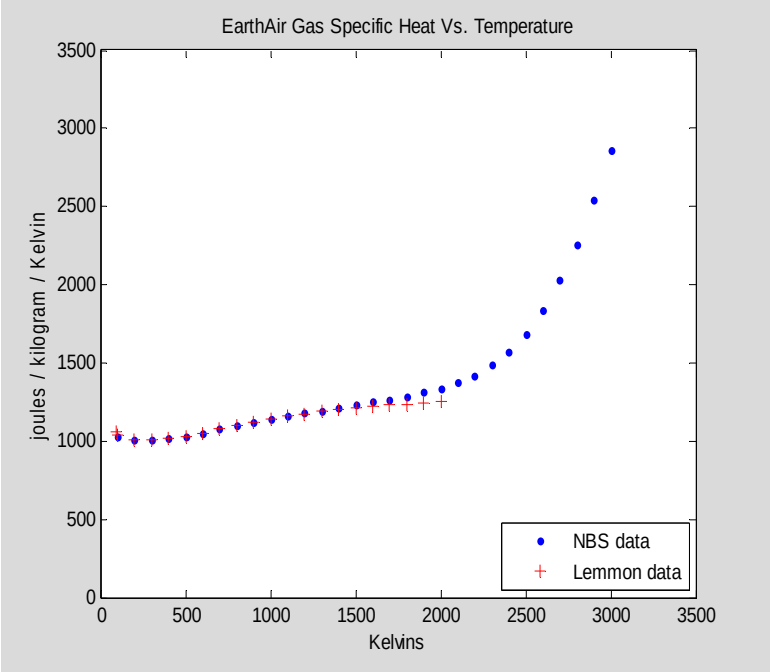
C.8.4 EarthAir Gas Specific Heat

EarthAir gas specific heat at constant pressure is taken from NBS Circular 564 and also from a recent paper by E. Lemmon. The NBS document explains that its tables include the effects of chemical disassociation above 1500 Kelvins, a fact which may be responsible for its increases in the specific heat for high temperature air which are not

seen in corresponding specific heat data for other gases taken from other references. It mentions that some elapsed time at high temperature is required for air's chemical disassociation to occur and cautions that the high temperature specific heat data should not be used for fast changes such as within shock waves. Apparently, the dat extracted here from Lemmon's paper do not include disassociation effects.

Specific heat values here are given in joules per mole per Kelvin and are tabulated vs temperature in Kelvins. The mass of a mole of EarthAir is `0.028966` kilogram, a fact used to convert specific heat to the joules per kilogram per Kelvin form plotted in the graph below.

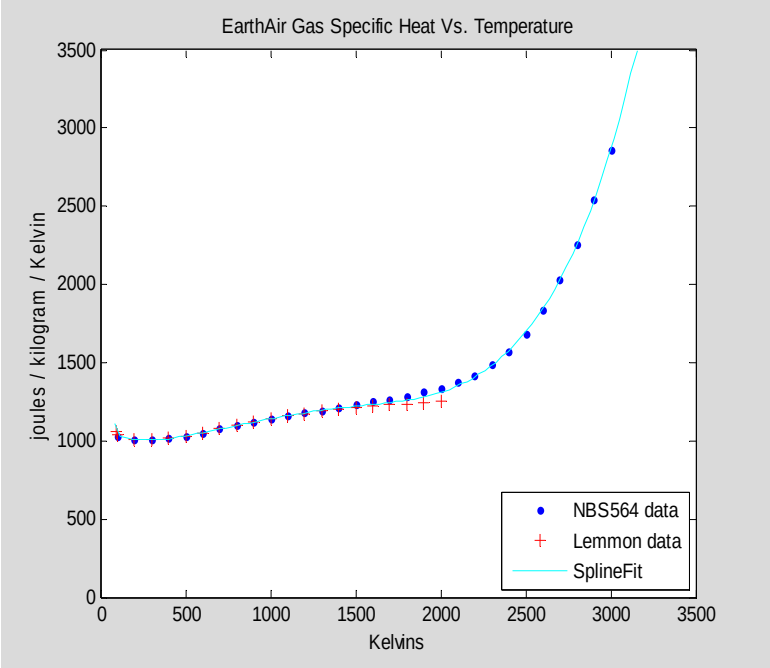
```
TK_Cp_EarthAirNBS=[100;200;;300;400;500;600;700;800;900;1000;1100;1200;
1300;1400;1500;1600;1700;1800;1900;2000;2100;2200;2300;2400;2500;2600;2
700;2800;2900;3000];
Cp_EarthAirNBS=287.041*[3.5824; 3.5062; 3.5059; 3.5333; 3.5882; 3.6626;
3.7455; 3.828; 3.906; 3.979; 4.046; 4.109; 4.171; 4.230; 4.289; 4.352;
4.418; 4.487; 4.566; 4.662; 4.781; 4.947; 5.179; 5.484; 5.882; 6.40;
7.06; 7.87; 8.86; 9.96];
TK_Cp_EarthAirLemmon=[90;100;200;;300;400;500;600;700;800;900;1000;1100
;1200;1300;1400;1500;1600;1700;1800;1900;2000];
Cp_EarthAirLemmon=[30.69; 30.13; 29.16; 29.15; 29.38; 29.83; 30.45;
31.14; 31.82; 32.47; 33.05; 33.57; 34.02; 34.42;34.77; 35.08; 35.35;
35.60; 35.82; 36.03; 36.21]/0.028966;
plot(TK_Cp_EarthAirNBS,Cp_EarthAirNBS,'.b',TK_Cp_EarthAirLemmon,Cp_Eart
hAirLemmon,'+r'),xlabel('Kelvins'),
ylim([0 3500])
xlim([0 3500])
ylabel('joules / kilogram / Kelvin'),
title('EarthAir Gas Specific Heat Vs. Temperature')
legend('NBS data','Lemmon data','Location','SouthEast')
```



Next, we generate a spline approximation model fitted to the data.

```
knots=augknt([0 100 500 1000 1500 2000 2500 3000],4);
Cp_EarthAir_pp=fn2fm(spap2(knots,4,
[TK_Cp_EarthAirNBS;TK_Cp_EarthAirLemmon],
[Cp_EarthAirNBS;Cp_EarthAirLemmon]),'pp');
Cp_EarthAir_fit=fnval(Cp_EarthAir_pp,linspace(85,3500));
plot(TK_Cp_EarthAirNBS,Cp_EarthAirNBS,'.b',
TK_Cp_EarthAirLemmon,Cp_EarthAirLemmon,'+r',linspace(85,3500),
Cp_EarthAir_fit,'-c'),
ylim([0 3500])
xlim([0 3500])

xlabel('Kelvins'),
ylabel('joules / kilogram / Kelvin'),
title('EarthAir Gas Specific Heat Vs. Temperature'),
legend('NBS564 data','Lemmon data','SplineFit','Location','SouthEast'),
format long g,
display(Cp_EarthAir_pp),
Cp_EarthAir_pp.coefs(:,1),
Cp_EarthAir_pp.coefs(:,2),
Cp_EarthAir_pp.coefs(:,3),
Cp_EarthAir_pp.coefs(:,4)

Cp_EarthAir_pp =
      form: 'pp'
    breaks: [0 100 500 1000 1500 2000 2500 3000]
     coefs: [7x4 double]
    pieces: 7
     order: 4
       dim: 1
```

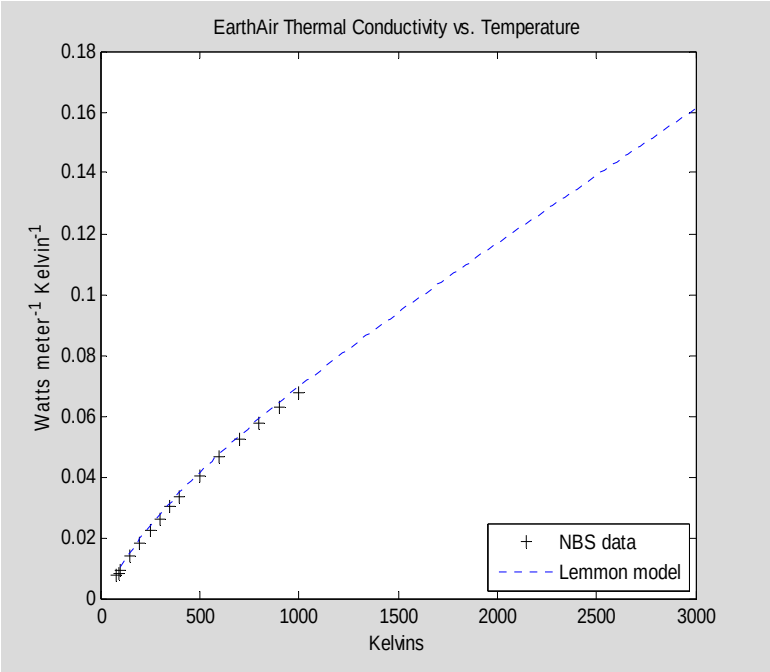
```
ans =
       -0.0219433024220399
    -1.02799909248604e-006
    -1.28735929876774e-007
     5.14388935689814e-009
     4.19542473208218e-007
     5.94054781577479e-007
     1.12868777500029e-006
ans =
          6.58432566100875
       0.00133493439678952
      0.000101335485806274
    -9.17684090088861e-005
    -8.40525749735412e-005
      0.000545261134838786
         0.001436343307205
ans =
          -658.940534692342
         -0.374475151788105
          0.200032801250212
          0.204816339648904
          0.116905847657693
          0.347510127590315
           1.33831234861221
ans =
          23027.7957894584
          1033.69650827185
          1031.70400912383
          1140.96228996591
          1221.07134370775
          1310.95393294424
          1695.28112814628
```



C.8.5 EarthAir Gas Thermal Conductivity

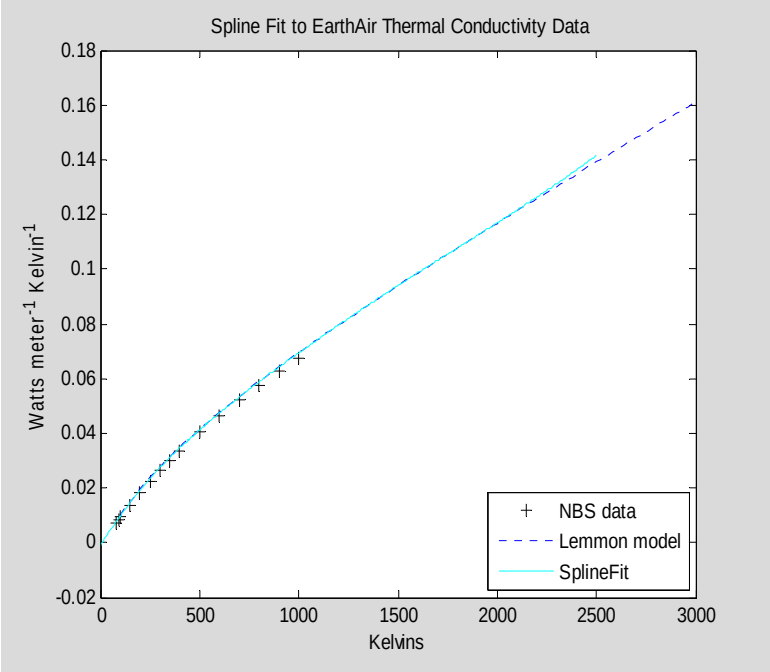
Thermal conductivity data for argon gas was found in NBS Circular 564. In addition, a 2004 paper by Lemmon et al provides comprehensive formulae for the thermal conductivity of (earth) air. Values here are given in watts per meter per Kelvin, and are tabulated versus temperature in Kelvins. Since the thermal conductivities of gases such as argon are almost independent of pressure in the range of pressures relevant to this work, no pressure dependence is modeled here.

```
TK_EarthAirThrmCondNBS=[80;90;100;150;200;250;300;350;400;500;600;700;8
00;900;1000];
EarthAirThrmCondNBS=0.02414*[0.3092; 0.3459; 0.3831; 0.5687; 0.7494;
0.9225; 1.087; 1.244; 1.394; 1.674; 1.931; 2.169; 2.392; 2.600; 2.798];
TK=(100:20:3000)';
EarthAirThrmCondLemmon=(1.308* 2.66958E-2*sqrt(28.9586*TK)/(0.360)^2
./exp(0.431-0.4623*log(TK/103.3) +0.08406*log(TK/103.3).^2
+0.005341*log(TK/103.3).^3-0.00331*log(TK/103.3).^4 )
+1.405*(TK/132.6312).^1.1)/1000;
plot(TK_EarthAirThrmCondNBS,EarthAirThrmCondNBS,'+k',
TK,EarthAirThrmCondLemmon,':b'),
xlabel('Kelvins'),
ylabel('Watts meter^-^1 Kelvin^-^1'),
title('EarthAir Thermal Conductivity vs. Temperature')
legend('NBS data','Lemmon model','Location','SouthEast')
```



Next we approximate this data, including the extrapolated model points, by using least squares techniques to fit cubic splines with a priori specified knot sequences.

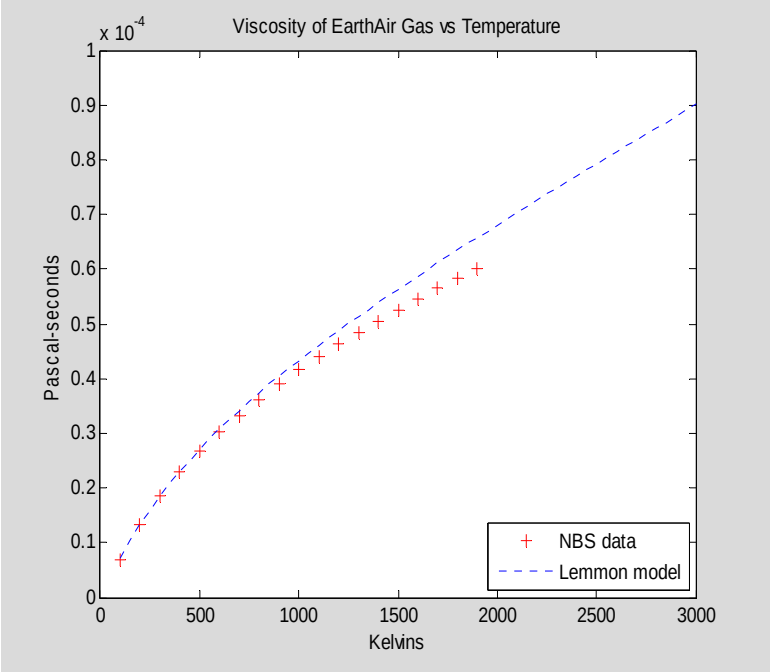
```
knots=augknt([0 500 1000 1500 2000],4);
kth_EarthAir_pp=fn2fm(spap2(knots,4,
[TK_EarthAirThrmCondNBS;TK], [EarthAirThrmCondNBS;
EarthAirThrmCondLemmon]),'pp');
kth_EarthAir_fit=fnval(kth_EarthAir_pp,linspace(000,2500,500));
plot(TK_EarthAirThrmCondNBS,EarthAirThrmCondNBS,'+k',TK,EarthAirThrmCon
dLemmon,':b',linspace(0,2500,500),kth_EarthAir_fit,'-c'),
xlabel('Kelvins'),
ylabel('Watts meter^-^1 Kelvin^-^1'),
title('  Spline Fit to EarthAir Thermal Conductivity Data '),
legend('NBS data','Lemmon model','SplineFit','Location','SouthEast'),
format long g,
display(kth_EarthAir_pp),
kth_EarthAir_pp.coefs(:,1),
kth_EarthAir_pp.coefs(:,2),
kth_EarthAir_pp.coefs(:,3),
kth_EarthAir_pp.coefs(:,4)

kth_EarthAir_pp =
      form: 'pp'
    breaks: [0 500 1000 1500 2000]
     coefs: [4x4 double]
    pieces: 4
     order: 4
       dim: 1
ans =
     4.33404489204219e-011
     1.05934327817733e-011
    -4.64891217598021e-013
     5.75818216393941e-012
ans =
     -8.5179120594456e-008
    -2.01684472138231e-008
    -4.27829804116303e-009
    -4.97563486756013e-009
ans =
      0.000116367375504442
     6.36935916003023e-005
     5.14702189728091e-005
     4.68432525184476e-005
ans =
      -0.00106933943641563
         0.041237124282244
        0.0693659873766611
        0.0939734109505751
```



C.8.6 EarthAir Gas Viscosity

Viscosity data for EarthAir gas was found in tabular form in NBS Circular 564, and in formula form in a 2004 paper by Lemmon et al. Values here are given in Pascal-seconds, and are tabulated versus temperature in Kelvins. Since the viscosities of gases such as argon are almost independent of pressure in the range of pressures relevant to this work, no pressure dependence is modeled here.

```
TK_EarthAirViscNBS=[100;200;300;400;500;600;700;800;900;1000;1100;1200;
1300;1400;1500;1600;1700;1800;1900];
EarthAirViscNBS=1.716e-5*[0.4038; 0.7742; 1.076; 1.332; 1.556; 1.758;
1.942; 2.112; 2.271; 2.420; 2.562; 2.696; 2.824; 2.947; 3.066; 3.180;
3.290; 3.397; 3.501];
TK=(100:50:3000)';
EarthAirViscLemmon=2.66958E-8*sqrt(28.9586*TK)/(0.360)^2./exp(0.431-
0.4623*log(TK/103.3)+0.08406*log(TK/103.3).^2+0.005341*log(TK/103.3).^3
-0.00331*log(TK/103.3).^4);
plot(TK_EarthAirViscNBS,EarthAirViscNBS,'+r',
TK,EarthAirViscLemmon,':b'),
title('Viscosity of EarthAir Gas vs Temperature'),
ylabel('Pascal-seconds'),
xlabel('Kelvins'),
legend('NBS data','Lemmon model','Location','SouthEast')
```

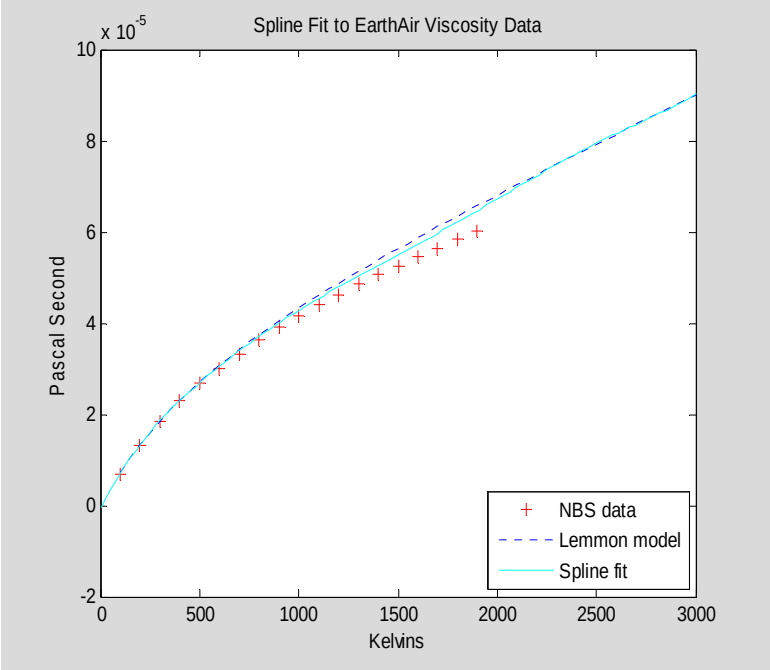


Next we approximate this data by using least squares techniques to fit a cubic spline with knot sequences specified a priori.

```
knots=augknt([0 500 1000 1500 2000 2500 3000],4);
mu_EarthAir_pp=fn2fm(spap2(knots,4,[TK_EarthAirViscNBS; TK],
[EarthAirViscNBS; EarthAirViscLemmon]),'pp');
mu_EarthAir_fit=fnval(mu_EarthAir_pp,linspace(0,3000));
plot(TK_EarthAirViscNBS,EarthAirViscNBS,'+r',TK,EarthAirViscLemmon,':b'
,linspace(0,3000),mu_EarthAir_fit,'-c')
xlabel('Kelvins'),
ylabel('Pascal Second'),
title(' Spline Fit to EarthAir Viscosity Data '),
legend('NBS data','Lemmon model','Spline fit','Location','SouthEast')
format long g,
display(mu_EarthAir_pp),
mu_EarthAir_pp.coefs(:,1),
mu_EarthAir_pp.coefs(:,2),
mu_EarthAir_pp.coefs(:,3),
mu_EarthAir_pp.coefs(:,4)
```

```
mu_EarthAir_pp =
      form: 'pp'
    breaks: [0 500 1000 1500 2000 2500 3000]
     coefs: [6x4 double]
    pieces: 6
     order: 4
       dim: 1
ans =
     4.62055101087152e-014
     2.44033392827618e-015
```

```
     7.07218511033621e-015
     1.87071169061818e-016
    -8.13805121176328e-015
     1.97452839499109e-014
ans =
    -8.16494844583416e-011
    -1.23412192952689e-011
    -8.68071840285467e-012
     1.92755926264958e-012
     2.20816601624222e-012
    -9.99891080140277e-012
ans =
     8.43542817634652e-008
       3.735892988666e-008
     2.68479610375982e-008
     2.34713814674957e-008
     2.55392441069417e-008
     2.16438717143615e-008
ans =
    -6.56284588420321e-007
     2.68841739423163e-005
     4.27833758028636e-005
      5.4921199859741e-005
      6.7162164305284e-005
      7.9466571461345e-005
```

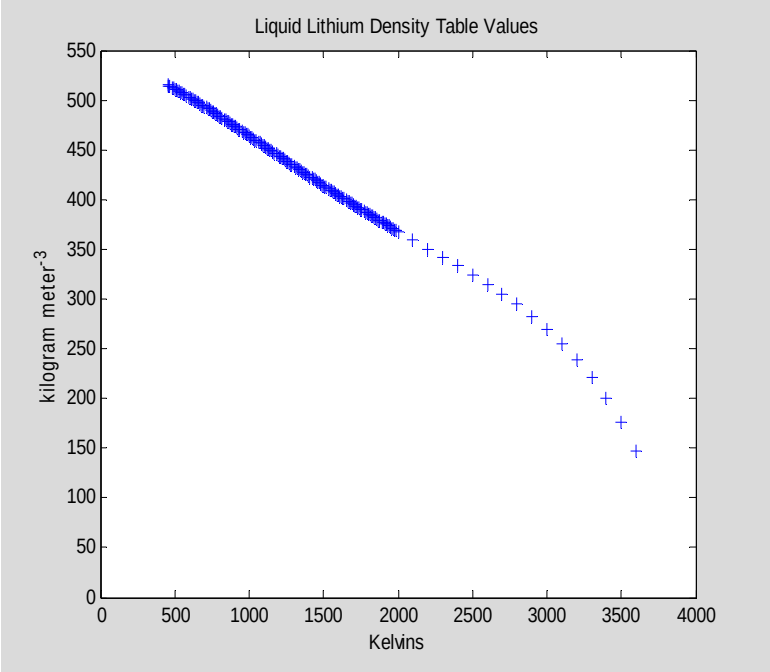


C.9 Lithium Liquid Properties

C.9.1 Lithium Liquid Density

Recommended reference data on density and thermal expansion of liquid metals compiled by Spil'rain et al of the Institute of High Temperatures, USSR Academy of Sciences, Moscow was published in a 1985 English translation as chapter 6.3.3 of the Handbook of Thermodynamic and Transport Propeties of Alkali Metals, edited by R. Ohse of Karlsruhe [Ohse 1985].

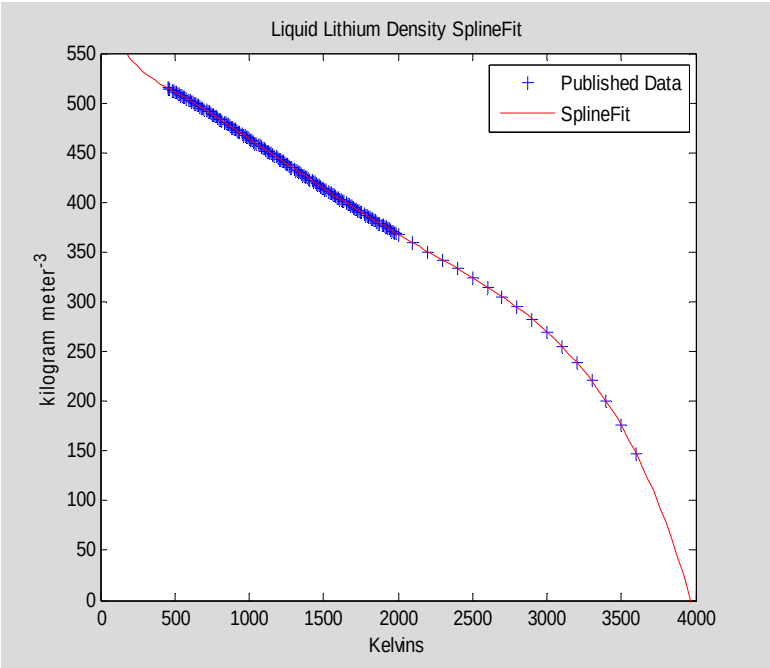
```
TK_DensityLi=[453.69;(460:10:2000)';(2100:100:3600)'];
DensityLi=[0.5147;0.5142; 0.5134; 0.5120; 0.5118; 0.5110; 0.5102;
0.5093; 0.5085; 0.5076; 0.5068; 0.5059; 0.5050; 0.5042; 0.5033; 0.5024;
0.5015; 0.5006; 0.4997; 0.4988; 0.4979; 0.4970; 0.4961; 0.4951; 0.4942;
0.4933; 0.4923; 0.4914; 0.4905; 0.4895; 0.4886; 0.4876; 0.4866; 0.4857;
0.4847; 0.4838; 0.4828; 0.4818; 0.4808; 0.4799; 0.4789; 0.4779; 0.4769;
0.4759; 0.4749; 0.4739; 0.4729; 0.4719; 0.4709; 0.4699; 0.4689; 0.4679;
0.4669; 0.4659; 0.4649; 0.4639; 0.4629; 0.4619; 0.4609; 0.4599; 0.4589;
0.4579; 0.4569; 0.4558; 0.4548; 0.4538; 0.4528; 0.4518; 0.4508; 0.4498;
0.4488; 0.4477; 0.4467; 0.4457; 0.4447; 0.4437; 0.4427; 0.4417; 0.4407;
0.4397; 0.4387; 0.4377; 0.4367; 0.4356; 0.4346; 0.4336; 0.4326; 0.4316;
0.4306; 0.4296; 0.4287; 0.4277; 0.4267; 0.4257; 0.4247; 0.4237; 0.4227;
0.4217; 0.4207; 0.4198; 0.4188; 0.4178; 0.4168; 0.4158; 0.4149; 0.4139;
0.4129; 0.4120; 0.4110; 0.4100; 0.4091; 0.4081; 0.4071; 0.4062; 0.4052;
0.4043; 0.4033; 0.4024; 0.4014; 0.4005; 0.3996; 0.3986; 0.3977; 0.3967;
0.3958; 0.3949; 0.3939; 0.3930; 0.3921; 0.3912; 0.3903; 0.3893; 0.3884;
0.3875; 0.3866; 0.3857; 0.3848; 0.3839; 0.3830; 0.3821; 0.3812; 0.3803;
0.3794; 0.3785; 0.3776; 0.3767;0.3758; 0.3749; 0.3740; 0.3731; 0.3723;
0.3714; 0.3705; 0.3696; 0.3687; 0.3679; 0.3592; 0.3505; 0.3419; 0.3332;
0.3242; 0.3149; 0.3050; 0.2943; 0.2827; 0.2698; 0.2555; 0.2392; 0.2208;
0.1996; 0.1754; 0.1476]*1e3;
plot(TK_DensityLi,DensityLi,'+b'),
ylim([0 550])
xlim([0 4000])
xlabel('Kelvins')
ylabel('kilogram meter^-^3')
title('Liquid Lithium Density Table Values')
```



This data is approximated by using least squares techniques to fit a cubic spline with knot sequences specified a priori.

```
knots=augknt([0 500 1000 1500 2000 2500 3000 3500 4000],4);
DensityLi_pp=fn2fm(spap2(knots,4,TK_DensityLi, DensityLi),'pp');
DensityLi_fit=fnval(DensityLi_pp,linspace(0,4000));
plot(TK_DensityLi,DensityLi,'+b',linspace(0,4000),DensityLi_fit,'r'),
ylim([0 550])
xlim([0 4000])
xlabel('Kelvins')
ylabel('kilogram meter^-^3')
title('Liquid Lithium Density SplineFit')
legend('Published Data','SplineFit','Location','NorthEast')
display(DensityLi_pp),
DensityLi_pp.coefs(:,1),
DensityLi_pp.coefs(:,2),
DensityLi_pp.coefs(:,3),
DensityLi_pp.coefs(:,4)

DensityLi_pp =
      form: 'pp'
    breaks: [0 500 1000 1500 2000 2500 3000 3500 4000]
     coefs: [8x4 double]
    pieces: 8
     order: 4
       dim: 1
ans =
    -4.74123774637388e-007
     2.07512438487717e-008
```

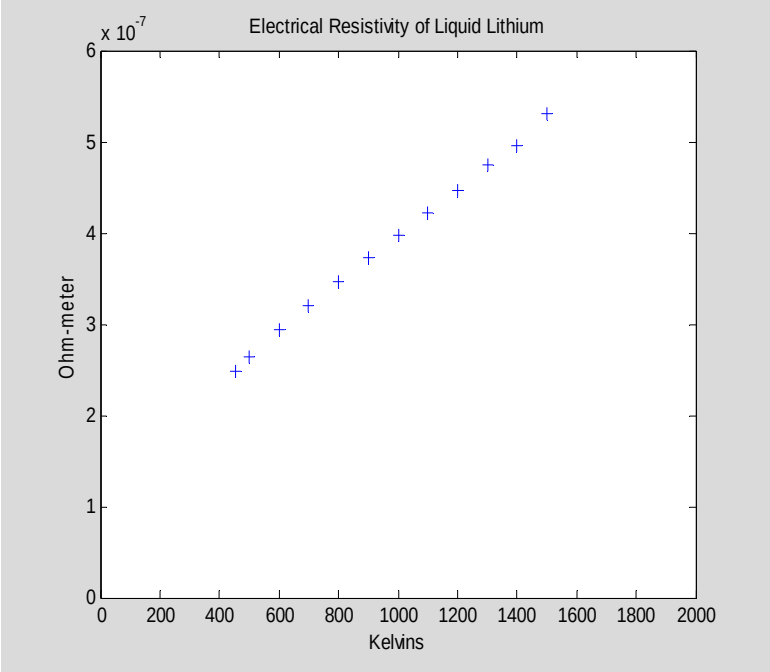
```
     9.12939440138192e-009
    -1.36927558974785e-009
    -1.64788928583478e-008
    -3.75494562756898e-008
     -6.7797307153971e-008
    -1.53394798033876e-007
ans =
      0.000676951392716666
    -3.42342692394153e-005
    -3.10740346625868e-006
     1.05866881358141e-005
     8.53277475119285e-006
    -1.61855645363285e-005
    -7.25097489498628e-005
     -0.000174205709680819
ans =
        -0.40344700761534
       -0.0820884458767153
        -0.100759282229552
       -0.0970196398947739
        -0.087459908451271
       -0.0912863033438391
        -0.135633960086935
        -0.258991689402276
ans =
         602.663185764455
         510.912058306278
         463.903173539162
         413.887855857995
         367.853548495843
         324.196926350712
         269.813701510249
          175.39462083507
```



C.9.2 Lithium Liquid Electrical Resistivity

Recommended values for the electrical resistivity of liquid lithium were obtained from [Ohse 1985, 730].

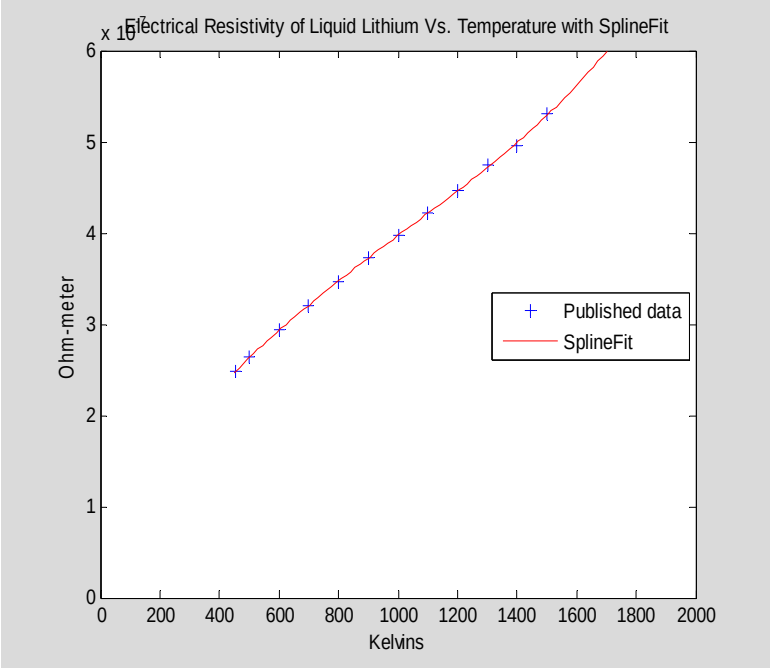
```
TK_LiResist=[453.7; (500:100:1500)'];
LiResist=1e-8*[24.80; 26.33; 29.34; 32.10; 34.71; 37.22; 39.69; 42.13;
44.61; 47.41; 49.57; 53.00];
plot(TK_LiResist,LiResist,'+b')
xlim([0 2000])
ylim([0 6e-7])
xlabel('Kelvins')
ylabel('Ohm-meter')
title('Electrical Resistivity of Liquid Lithium')
```



This data is approximated by using least squares techniques to fit a cubic spline to the data with knot sequences specified a priori.

```
knots=augknt([0 500 1000 1500],4);
LiResist_pp=fn2fm(spap2(knots,4,TK_LiResist, LiResist),'pp');
LiResist_fit=fnval(LiResist_pp,linspace(450,2000));
plot(TK_LiResist,LiResist,'+b',linspace(450,2000),LiResist_fit,'r'),
xlim([0 2000])
ylim([0 6e-7])
ylabel('Ohm-meter')
xlabel('Kelvins')
```

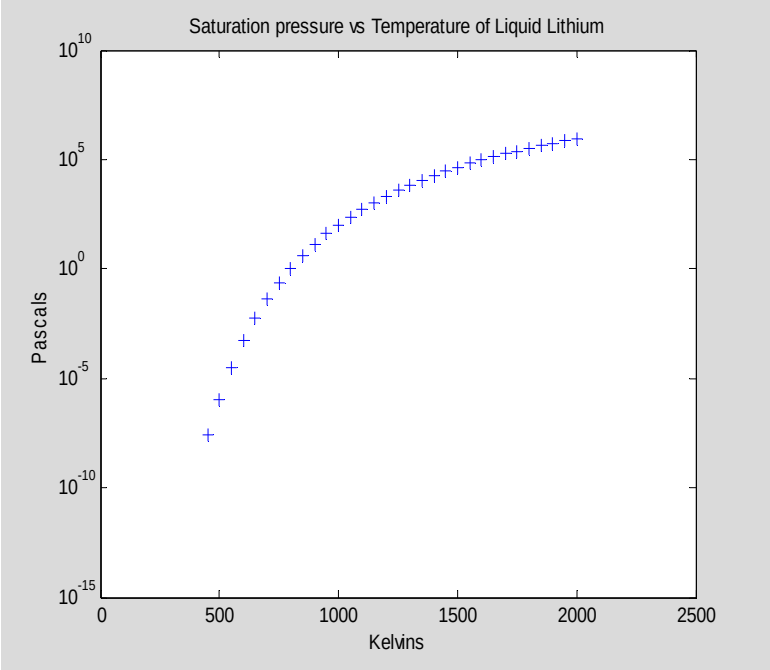
```
title('Electrical Resistivity of Liquid Lithium Vs. Temperature with
SplineFit'),
legend('Published data','SplineFit','Location','East')
LiResist_pp
LiResist_pp.coefs(:,1)
LiResist_pp.coefs(:,2)
LiResist_pp.coefs(:,3)
LiResist_pp.coefs(:,4)

LiResist_pp =
      form: 'pp'
    breaks: [0 500 1000 1500]
     coefs: [3x4 double]
    pieces: 3
     order: 4
       dim: 1
ans =
     1.17129579300351e-014
     5.52903436368389e-017
     1.21321932975434e-016
ans =
    -1.76726892761036e-011
    -1.03252381050998e-013
    -2.03168655957388e-014
ans =
     9.19304605019126e-009
     3.05075221613967e-010
     2.43290598290598e-010
ans =
    -1.37896157904531e-006
     2.63508868278809e-007
     3.97144676777648e-007
```



C.9.3 Liquid Lithium Saturation Pressure

Recommended values for liquid lithium's saturation pressure vs. temperature were taken from [Ohse 1995, 505].

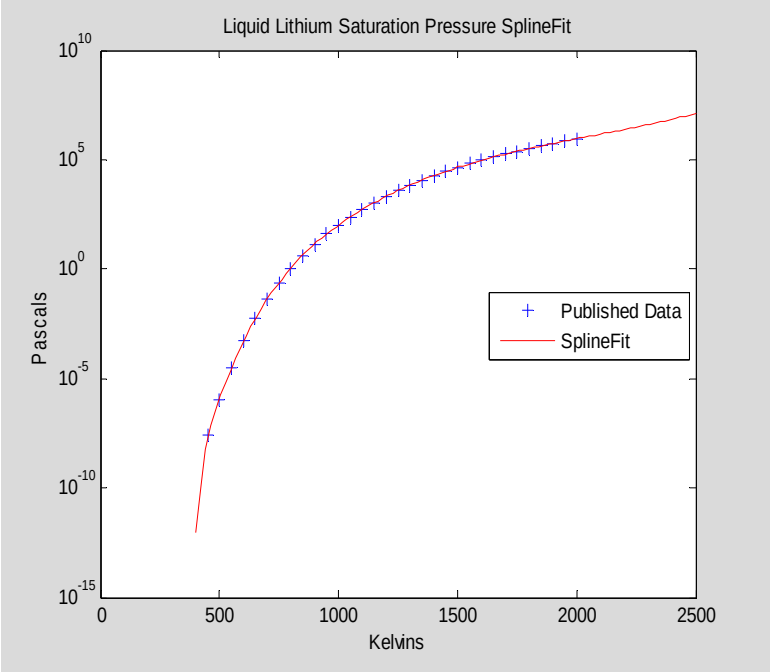
```
TK_LiPsat=[454;(500:50:2000)'];
LiPsat=1e6*[.24706e-13; .10710e-11; .30644e-10; .49993e-9; .52954e-8;
.39955e-7; .22986e-6; .10610e-5; .40851e-5; .13525e-4; .39432e-4;
.10320e-3; .24622e-3; .54238e-3;  .11147e-2; .21558e-2;  .39528e-2;
.69134e-2; .11595e-1; .18732e-1; .29265e-1; .44361e-1; .65438e-1;
.94178e-1; .13253; .18274; .24731; .32902; .43090; .55622; .70848;
.89135];
semilogy(TK_LiPsat,LiPsat,'+b')
xlim([0 2500])
ylim([1e-15 1e10])
ylabel('Pascals')
xlabel('Kelvins')
title('Saturation pressure vs Temperature of Liquid Lithium')
```



This data is approximated by using least squares techniques to fit a cubic spline ***to its logarithm*** with knot sequences specified a priori.

```
knots=augknt([0 500 1000 1500 2000],4);
```

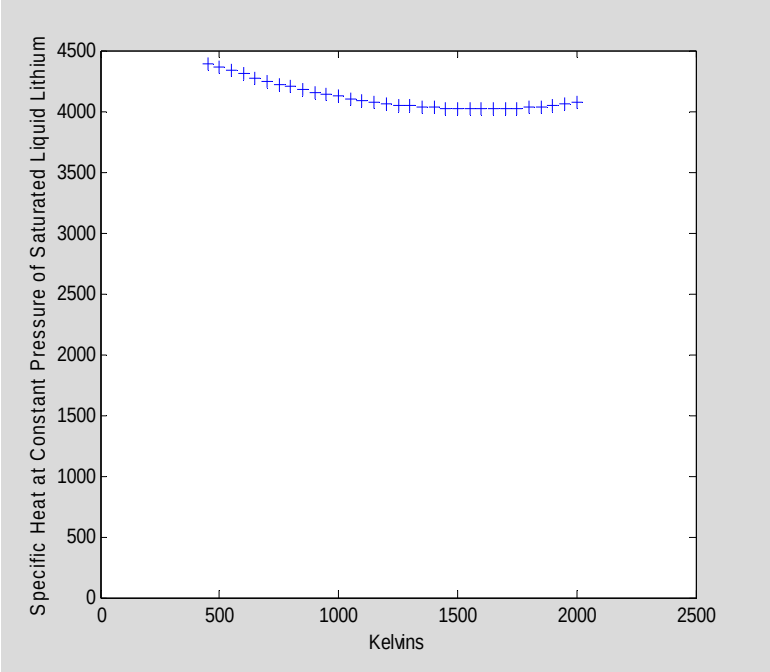
```
LiPsat_pp=fn2fm(spap2(knots,4,TK_LiPsat, log(LiPsat)),'pp');
LiPsat_fit=exp(fnval(LiPsat_pp,linspace(400,2500)));
semilogy(TK_LiPsat,LiPsat,'+b',linspace(400,2500),LiPsat_fit,'r'),
xlim([0 2500])
ylim([1e-15 1e10])
xlabel('Kelvins')
ylabel('Pascals')
title('Liquid Lithium Saturation Pressure SplineFit')
legend('Published Data','SplineFit','Location','East')
display(LiPsat_pp),
LiPsat_pp.coefs(:,1),
LiPsat_pp.coefs(:,2),
LiPsat_pp.coefs(:,3),
LiPsat_pp.coefs(:,4)

LiPsat_pp =
      form: 'pp'
    breaks: [0 500 1000 1500 2000]
     coefs: [4x4 double]
    pieces: 4
     order: 4
       dim: 1
ans =
     6.50398580730858e-006
     4.74527398156998e-008
     5.34608932461358e-009
     3.35170306099235e-009
ans =
      -0.00984088880271437
    -8.49100917515006e-005
    -1.37309820279509e-005
    -5.71184804103053e-006
ans =
          5.02995389193575
        0.0670544447028205
        0.0177339078130948
       0.00801249277860416
ans =
         -881.371944737673
         -13.6189735348157
          4.61231835568192
          10.7147879208183
```



C.9.4 Lithium Liquid Specific Heat

Recommended values for liquid lithium specific heat at constant pressure vs. temperature at saturated pressure condictions were taken from [Ohse 1985, 505].

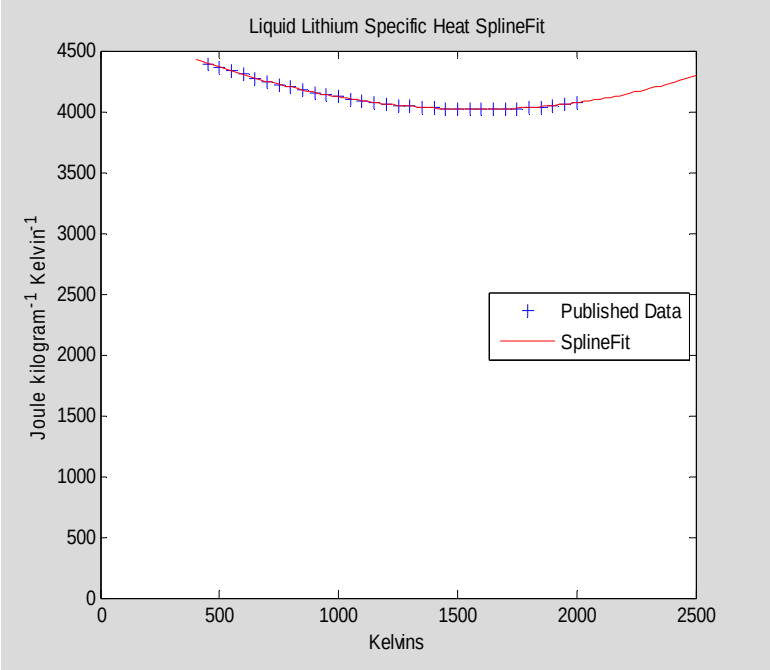
```
TK_LiCp=[454;(500:50:2000)'];
LiCp=1e3*[ .4393e1; .4363e1; .4332e1; .4303e1; .4275e1; .4248e1;
.4223e1; .4199e1; .4177e1; .4156e1; .4137e1; .4119e1; .4102e1; .4087e1;
.4074e1; .4062e1; .4051e1; .4042e1; .4034e1; .4028e1; .4023e1; .4020e1;
.4018e1; .4017e1; .4019e1; .4021e1; .4026e1; .4031e1; .4039e1; .4048e1;
.4059e1; .4071e1 ];
plot(TK_LiCp,LiCp,'+b')
xlim([0 2500])
ylim([0 4500])
xlabel('Kelvins')
ylabel('Specific Heat at Constant Pressure of Saturated Liquid
Lithium')
```



This data is approximated by using least squares techniques to fit a cubic spline to the data with knot sequences specified a priori.

```
knots=augknt([0 500 1000 1500 2000],4);
LiCp_pp=fn2fm(spap2(knots,4,TK_LiCp, LiCp),'pp');
LiCp_fit=fnval(LiCp_pp,linspace(400,2500));
plot(TK_LiCp,LiCp,'+b',linspace(400,2500),LiCp_fit,'r'),
xlim([0 2500])
ylim([0 4500])
xlabel('Kelvins')
ylabel('Joule kilogram^-^1 Kelvin^-^1')
title('Liquid Lithium Specific Heat SplineFit')
legend('Published Data','SplineFit','Location','East')
display(LiCp_pp),
LiCp_pp.coefs(:,1),
LiCp_pp.coefs(:,2),
LiCp_pp.coefs(:,3),
LiCp_pp.coefs(:,4)

LiCp_pp =
      form: 'pp'
    breaks: [0 500 1000 1500 2000]
     coefs: [4x4 double]
    pieces: 4
     order: 4
       dim: 1
ans =
    -3.48687295322885e-006
     6.94329072917594e-009
    -2.63017784945918e-010
```

```
     3.60799763081086e-008
ans =
       0.00551209812493144
      0.000281788695088167
      0.000292203631181927
      0.000291809104504517
ans =
         -3.52802041889946
        -0.631077008889664
        -0.344080845754615
       -0.0520744779113993
ans =
          5184.87959181697
          4363.03479444649
          4118.81137511485
          4019.78898280991
```

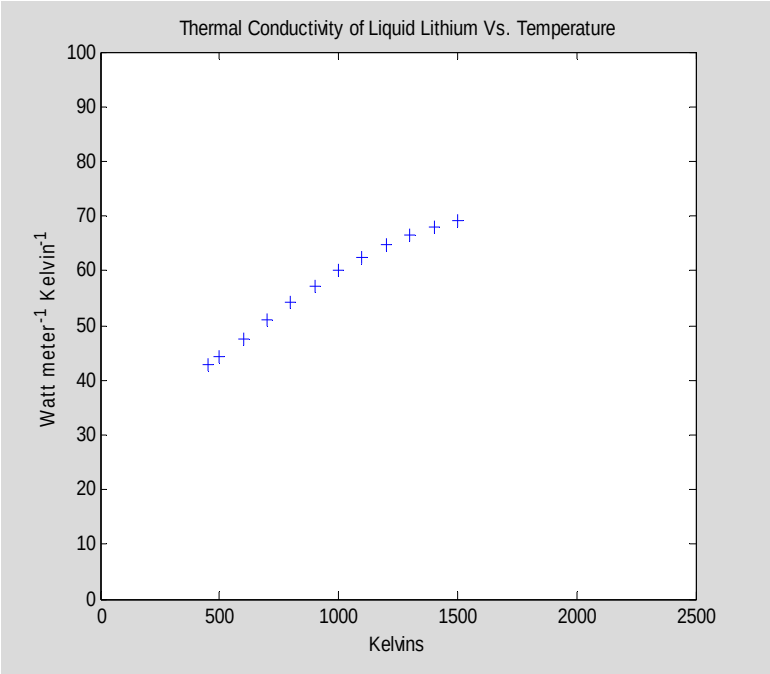


C.9.5 Lithium Liquid Thermal Conductivity

Recommended values for thermal conductivity of liquid lithium were taken from [Ohse1985, 750]

```
TK_LiThrmCond=[453.7; (500:100:1500)'];
LiThrmCond=[42.8; 44.3; 47.6; 50.9; 54.1; 57.2; 60.0; 62.5; 64.7; 66.5;
68.0; 69.1];
plot(TK_LiThrmCond, LiThrmCond,'+b')
xlim([0 2500])
```

```
ylim([0 100])
ylabel('Watt meter^-^1 Kelvin^-^1')
xlabel('Kelvins')
title('Thermal Conductivity of Liquid Lithium Vs. Temperature')
```

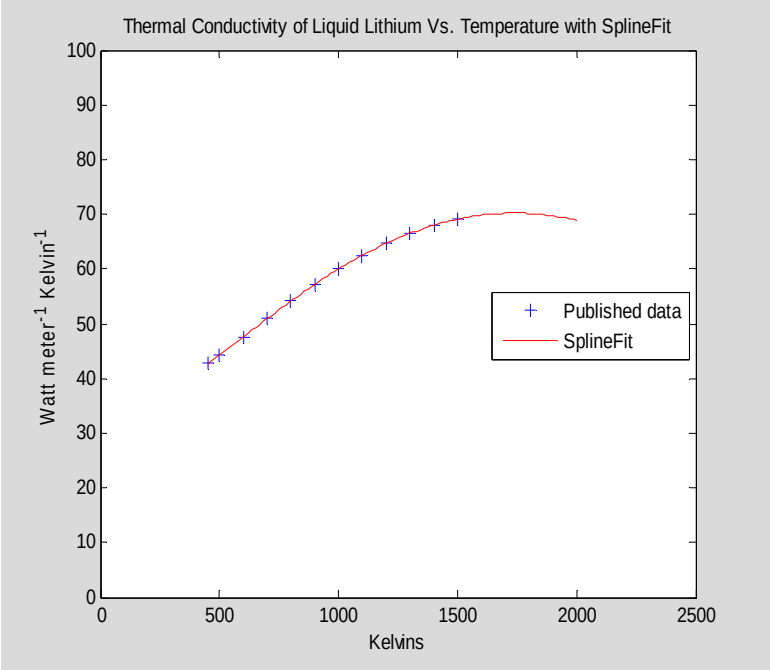


This data is approximated by using least squares techniques to fit a cubic spline to the data with knot sequences specified a priori.

```
knots=augknt([0 500 1000 1500],4);
LiThrmCond_pp=fn2fm(spap2(knots,4,TK_LiThrmCond, LiThrmCond),'pp');
LiThrmCond_fit=fnval(LiThrmCond_pp,linspace(450,2000));
plot(TK_LiThrmCond,LiThrmCond,'+b',linspace(450,2000),LiThrmCond_fit,'r
'),
xlim([0 2500])
ylim([0 100])
ylabel('Watt meter^-^1 Kelvin^-^1')
xlabel('Kelvins')
title('Thermal Conductivity of Liquid Lithium Vs. Temperature with
SplineFit'),
legend('Published data','SplineFit','Location','East')
LiThrmCond_pp
LiThrmCond_pp.coefs(:,1)
LiThrmCond_pp.coefs(:,2)
LiThrmCond_pp.coefs(:,3)
LiThrmCond_pp.coefs(:,4)
```

```
LiThrmCond_pp =
      form: 'pp'
    breaks: [0 500 1000 1500]
     coefs: [3x4 double]
```

```
    pieces: 3
     order: 4
       dim: 1
ans =
    -3.96780228307425e-009
      -1.373130078214e-008
    -1.65331460247586e-009
ans =
     1.03284191734894e-005
     4.37671574887792e-006
    -1.62202354243321e-005
ans =
        0.0252929686003196
        0.0326455360615033
        0.0267237762237763
ans =
          29.5690983740431
           44.301712182191
          60.0022465523946
```

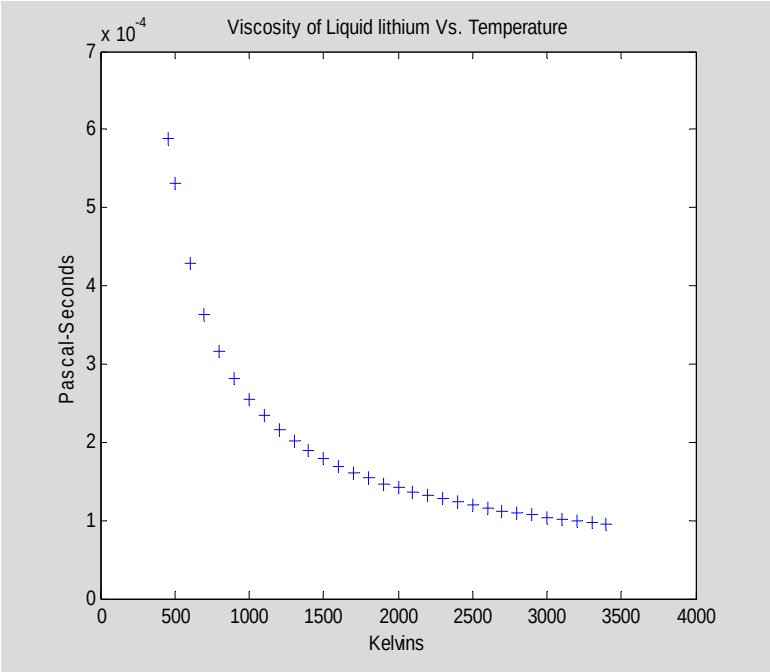


C.9.6 Lithium Liquid Viscosity

Recommended dynamic viscosity data for liquid lithium was taken from [Ohse 1985, 777-8].

```
TK_LiVisc=[460; (500:100:3400)'];
LiVisc=1e-5*[58.875; 53.062; 42.857; 36.236; 31.588; 28.138; 25.470;
23.340; 21.598; 20.143; 18.907; 17.844; 16.918; 16.102; 15.379; 14.731;
14.149; 13.620; 13.139; 12.698; 12.293; 11.919; 11.573; 11.251; 10.951;
10.670; 10.407; 10.160; 9.927; 9.707; 9.499];
```

```
plot(TK_LiVisc,LiVisc,'+b')
ylim([0 7e-4])
xlim([0 4000])
title('Viscosity of Liquid lithium Vs. Temperature')
xlabel('Kelvins')
ylabel('Pascal-Seconds')
```

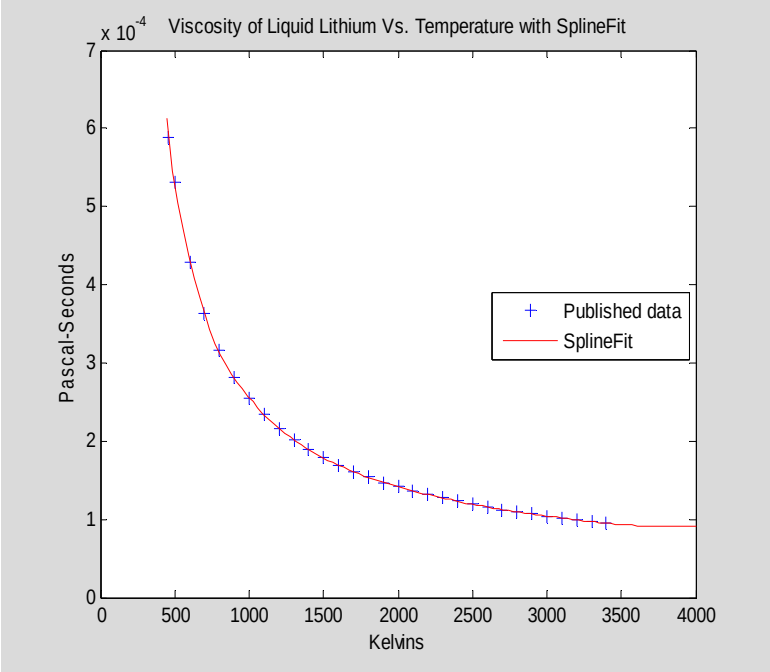


This data is approximated by using least squares techniques to fit a cubic spline to the data with knot sequences specified a priori.

```
knots=augknt([0 500 1000 1500 2000 2500 3000 3500],4);
LiVisc_pp=fn2fm(spap2(knots,4,TK_LiVisc, LiVisc),'pp');
LiVisc_fit=fnval(LiVisc_pp,linspace(450,4000));
plot(TK_LiVisc,LiVisc,'+b',linspace(450,4000),LiVisc_fit,'r'),
xlim([0 4000])
ylim([0 7e-4])
ylabel('Pascal-Seconds')
xlabel('Kelvins')
title('Viscosity of Liquid Lithium Vs. Temperature with SplineFit'),
legend('Published data','SplineFit','Location','East')
LiVisc_pp
LiVisc_pp.coefs(:,1)
LiVisc_pp.coefs(:,2)
LiVisc_pp.coefs(:,3)
LiVisc_pp.coefs(:,4)
```

```
LiVisc_pp =
      form: 'pp'
    breaks: [0 500 1000 1500 2000 2500 3000 3500]
```

```
     coefs: [7x4 double]
    pieces: 7
     order: 4
       dim: 1
ans =
   -1.95331309831702e-010
   -9.80388718870332e-013
   -5.77397340032479e-014
    -3.9437989606833e-014
    6.21108299029389e-016
   -1.02105094854462e-014
    1.03295377449395e-014
ans =
    2.94630688965853e-007
    1.63372421830003e-009
    1.63141139994535e-010
    7.65315389896636e-011
     1.7374554579414e-011
     1.8306217027958e-011
    2.99045279978884e-012
ans =
    -0.000149251629630418
   -1.11942303834102e-006
   -2.20990359193739e-007
    -1.0115401970164e-007
    -5.4200972917101e-008
    -3.6360587113415e-008
   -2.57122521995416e-008
ans =
       0.0259134142185961
      0.00052885791588785
     0.000255028861433555
     0.000178101500084913
     0.000141727626280655
     0.000119048417004336
     0.000104168364018938
```



C.10 M-File Subroutine Codes for Properties of All Investigated Fluids

C.10.1 Perfect Gas Constant M-file Subroutine Code

Gas density, temperature and pressure are related by the ideal gas law, i.e.,

$$\rho = \frac{p}{RT}$$

where R is the gas constant which depends on fluid type. An m-file for the gas constant follows:

```
function R=GasConstant(fluidtype)
% This returns the gas constant for recognized fluid types, in Joules/kilogram/Kelvin/
%  Recognized fluids include (gasses) 'MarsMix', 'Ar', 'N2','CO2','He','EarthAir'.)
if strcmpi(fluidtype,'MarsMix');
R=191.06037;
elseif strcmpi(fluidtype,'Ar');
R=208.1108;
elseif strcmpi(fluidtype,'CO2'),
R=188.9036;
elseif strcmpi(fluidtype,'N2');
R=296.7726;
elseif strcmpi(fluidtype,'He');
   R=2077.0524;
elseif strcmpi(fluidtype,'EarthAir');
R=287.0127;
end
```

C.10.2 Liquid Lithium Density M-file Subroutine Code

An m-file follows for liquid lithium density as a function of temperature.

```
function rhoLi=LiDensity(T)
% This returns liquid lithium density  (kg/m3) for temperatures T in Kelvins.
%
Tmp=453.69;
LiDensity_pp.form='pp';
LiDensity_pp.breaks=  [0 500 1000 1500 2000 2500 3000 3500 4000];
LiDensity_pp.pieces=8;
LiDensity_pp.order=4;
LiDensity_pp.dim=1;
LiDensity_pp.form='pp';
LiDensity_pp.coefs(:,1)=[    -4.74123774637388e-007
    2.07512438487717e-008
    9.12939440138192e-009
   -1.36927558974785e-009
   -1.64788928583478e-008
   -3.75494562756898e-008
    -6.7797307153971e-008
   -1.53394798033876e-007];
LiDensity_pp.coefs(:,2)=[      0.000676951392716666
   -3.42342692394153e-005
   -3.10740346625868e-006
    1.05866881358141e-005
    8.53277475119285e-006
   -1.61855645363285e-005
   -7.25097489498628e-005
   -0.000174205709680819];
LiDensity_pp.coefs(:,3)=[         -0.40344700761534
     -0.0820884458767153
      -0.100759282229552
     -0.0970196398947739
      -0.087459908451271
     -0.0912863033438391
      -0.135633960086935
      -0.258991689402276];
LiDensity_pp.coefs(:,4)=[          602.663185764455
         510.912058306278
         463.903173539162
         413.887855857995
         367.853548495843
         324.196926350712
         269.813701510249
          175.39462083507];
rhoLi=fnval(LiDensity_pp,T);
rhoLi(find(T<=Tmp))=NaN;
```

C.10.3 Lithium Liquid Electrical Resistivity M-file Subroutine Code

```
function etaLi=LiResistivity(T)
% This returns liquid lithium's electrical resistivity  (ohm-meters) for temperatures T in Kelvins.
%
Tmp=453.69;
LiResistivity_pp.form='pp';
LiResistivity_pp.breaks=  [0 500 1000 1500];
LiResistivity_pp.pieces=3;
LiResistivity_pp.order=4;
LiResistivity_pp.dim=1;
LiResistivity_pp.form='pp';
LiResistivity_pp.coefs(:,1)=[      1.17129579300351e-014
    5.52903436368389e-017
    1.21321932975434e-016  ];
LiResistivity_pp.coefs(:,2)=[     -1.76726892761036e-011
   -1.03252381050998e-013
   -2.03168655957388e-014 ];
```

```
LiResistivity_pp.coefs(:,3)=[      9.19304605019126e-009
    3.05075221613967e-010
    2.43290598290598e-010  ];
LiResistivity_pp.coefs(:,4)=[     -1.37896157904531e-006
    2.63508868278809e-007
    3.97144676777648e-007  ];
etaLi=fnval(LiResistivity_pp,T);
etaLi(find(T<=Tmp))=NaN;
```

C.10.4 Enthalpy M-file Subroutine Code

An m-file for enthalpy follows:

```
function H0=Enthalpy(fluidtype,T)
% This returns enthalpy (joules/kg) for temperatures T for recognized fluid types.
% Results should be accurate for 100 Kelvins<T<6000 Kelvins, & extrapolated beyond that.
%  Here, T is any array of temperatures in Kelvins, H0 is the resulting array of same size.
%  Recognized fluids include (gasses) 'MarsMix', 'Ar', 'N2','CO2','He','EarthAir', and 'Li' (liquid).
H0_pp.form='pp';
H0_pp.breaks=[0 100 500 1000 1500 2000 3000 4000 5000 6000];
H0_pp.pieces=9;
H0_pp.order=4;
H0_pp.dim=1;
if strcmpi(fluidtype,'MarsMix');
H0_pp.coefs(:,1)=[.0102274805269917
    -0.000261293627148073
    -0.000127890163699401
    -4.0491846917942e-005
   -2.00615876268502e-005
   -5.13909684719186e-006
   -1.20579512292617e-006
   -3.54058916457128e-007
    1.31268140418166e-006];
H0_pp.coefs(:,2)=[
       -2.44358425545878
        0.624659902638749
        0.311107550061062
        0.119272304511961
       0.0585345341350491
       0.0284421526947733
       0.0130248621531977
      0.00940747678442164
      0.00834530003504754];
H0_pp.coefs(:,3)=[806.998446424539
         625.106011142536
          999.41299222246
         1214.60291950897
         1303.50633883248
         1346.99468224739
         1388.46169709536
         1410.89403603297
         1428.64681285245];
H0_pp.coefs(:,4)=[-212761.983720458
         -146270.5011056
         186994.695636138
         748491.808800208
         1380549.86381794
         2044428.96831459
         3414726.70640956
         4815007.47053519
         6234954.92443613];
elseif strcmpi(fluidtype,'Ar');
H0_pp.coefs(:,1)=[-1.86769571596165e-005
     1.0498011444416e-007
```

```
    -8.33938341141523e-009
    -2.91051424869693e-009
     1.07676430364134e-008
    -6.38050747119223e-009
     4.17366920830394e-009
    -2.98394920120397e-009
     9.06716937295945e-009];
H0_pp.coefs(:,2)=[0.00548859512409308
    -0.000114492023791968
     1.14841135412007e-005
    -1.02496157569476e-006
    -5.39073294817172e-006
     1.07607316057283e-005
    -8.38079080745047e-006
     4.14021681717713e-006
    -4.81163078643476e-006];
H0_pp.coefs(:,3)=[519.826808036436
          520.364218346466
          520.323015182366
          520.328244758348
          520.325036911085
          520.327721910415
          520.330101851213
          520.325861277223
          520.325189863253];
H0_pp.coefs(:,4)=[-155126.664664063
         -103107.774866338
          105026.312475766
          365189.648672407
          625353.150996907
          885515.667724592
          1405847.76985914
          1926173.66458876
          2446500.68213359];
elseif strcmpi(fluidtype,'CO2');
H0_pp.coefs(:,1)=[0.0107326301178543
    -0.000275726019996787
    -0.000133701237860417
    -4.16462216642564e-005
    -2.07122355094936e-005
     -5.2765267781502e-006
    -1.24023079762947e-006
    -3.60503394604128e-007
     1.36239384044052e-006];
H0_pp.coefs(:,2)=[-2.56557035988582
         0.654218675470476
         0.323347451474332
         0.122795594683707
        0.0603262621873232
        0.0292579089230821
        0.0134283285886315
       0.00970763619574564
       0.00862612601193041];
H0_pp.coefs(:,3)=[812.060914924536
          620.925746483001
          1011.95219726092
          1235.02372033994
          1326.58464877546
          1371.37673433066
          1414.06297184238
          1437.19893662675
          1455.53269883443];
H0_pp.coefs(:,4)=[-212770.791959973
         -146487.773948523
          188911.04744016
          759011.354206653
          1402016.33533952
          2077801.19583539
```

```
          3473159.31231099
          4899410.38194436
          6345956.45137225];
elseif strcmpi(fluidtype,'N2');
H0_pp.coefs(:,1)=[-0.00269638862197063
     0.000160833720360164
    3.88413102314841e-006
    -3.9770887585245e-005
   -1.50168106641709e-005
   -5.58052792502466e-006
    -1.1879336664623e-006
   -5.61037574534112e-007
    5.39105331688233e-007];
H0_pp.coefs(:,2)=[ 0.719899180225285
      -0.0890174063659046
       0.103983058066292
       0.109809254601013
       0.0501529232231483
       0.0276277072268938
       0.0108861234518217
      0.00732232245243768
      0.00563920972883602];
H0_pp.coefs(:,3)=[ 988.801689048395
         1051.88986643433
         1057.87612711449
         1164.77228344814
         1244.75337236022
         1283.64368758524
         1322.15751826395
         1340.36596416821
         1353.32749634948];
H0_pp.coefs(:,4)=[-309494.741802138
        -206111.969717016
         210694.549941223
         766113.894392933
          1370980.9888191
         2004018.80447198
         3309709.67135908
          4641565.3794084
         5988692.62845451];
elseif strcmpi(fluidtype,'He');
H0_pp.coefs(:,1)=[
    -0.000186405607513449
    1.04775536214561e-006
    -8.3231321854934e-008
   -2.90484242341336e-008
    1.07466597637944e-007
   -6.36807356077043e-008
    4.16553583787997e-008
   -2.97813427846449e-008
    9.04949987564881e-008];
H0_pp.coefs(:,2)=[
    -0.000186405607513449
    1.04775536214561e-006
    -8.3231321854934e-008
   -2.90484242341336e-008
    1.07466597637944e-007
   -6.36807356077043e-008
    4.16553583787997e-008
   -2.97813427846449e-008
    9.04949987564881e-008];
H0_pp.coefs(:,3)=[          5188.1380421324
         5193.50167253903
         5193.09044383779
         5193.14263768713
         5193.11062172689
         5193.13741939671
         5193.16117242599
```

```
        5193.11884932355
        5193.11214826793];
H0_pp.coefs(:,4)=[        -1548243.64163294
       -1029068.45309561
        1048216.44200817
        3644779.91434701
        6241345.04472703
        8837900.33834558
        14031081.4746247
        19224200.6578203
        24417331.0472875];
elseif strcmpi(fluidtype,'EarthAir');
H0_pp.breaks=[0 100 500 1000 1500 2000 3000]
H0_pp.pieces=6;
H0_pp.order=4;
H0_pp.dim=1;
H0_pp.coefs(:,1)=[ 0.027864725845585
      0.00025695302835123
   -4.94037917273004e-005
    1.73550880724561e-005
    7.74163690522254e-006
     0.000389207628736484];
H0_pp.coefs(:,2)=[  -8.52127015766061
       -0.161852403985123
        0.146491230036354
       0.0723855424454055
       0.0984181745540859
        0.110030629911925];
H0_pp.coefs(:,3)=[   1903.23433174482
        1034.92207558025
        1028.77760600074
        1138.21599224162
        1223.61785074137
        1327.84225297437];
H0_pp.coefs(:,4)=[   -34955.3138359624
        98020.1436074985
        502537.583016457
          1047373.71956
        1636747.48730122
        2274128.66092358];
end
H0=fnval(H0_pp,T);
```

C.10.5 Entropy Integral M-file Subroutine Code

An m-file for the entropy integral follows:

```
function S0=EntropyIntegral(fluidtype,T)
% This returns the entropy integral (joules/kg/Kelvin) for temperatures T for recognized fluid types.
% Results should be accurate for 100 Kelvins<T<6000 Kelvins, & extrapolated beyond that.
%  Here, T is any array of temperatures in Kelvins, S0 is the resulting array of same size.
%  Recognized fluids include (gases) 'MarsMix', 'Ar', 'N2','CO2','He', and 'EarthAir'.
S0_pp.form='pp';
S0_pp.breaks=[0 100 500 1000 1500 2000 3000 4000 5000 6000];
S0_pp.pieces=9;
S0_pp.order=4;
S0_pp.dim=1;
if strcmpi(fluidtype,'MarsMix');
S0_pp.coefs(:,1)=[ 0.003485573924265
    5.29285604313321e-006
    2.86919972067194e-007
     1.7787566394139e-007
    5.21139189360539e-008
```

```
    2.71444966661034e-008
    8.21911815563496e-009
     5.5373104734041e-009
    1.83880168983344e-009];
S0_pp.coefs(:,2)=[
        -1.0529469881161
      -0.0072748108366035
    -0.000923383584843663
    -0.000493003626742864
    -0.000226190130830779
    -0.000148019252426702
   -6.65857624283913e-005
   -4.19284079614855e-005
   -2.53164765412724e-005];
S0_pp.coefs(:,3)=[          111.232409185315
         5.21022929004383
         1.93095152146498
         1.22275791567171
        0.863161036884889
         0.67605634525615
        0.461451330401057
        0.352937160011179
        0.285692275508419];
S0_pp.coefs(:,4)=[                        0
         4079.34496163542
         5338.20973055692
         6108.70459158689
          6619.0671007297
         7000.61432633146
         7555.79591582701
         7958.88060195531
         8275.42666447841];
elseif strcmpi(fluidtype,'Ar');
S0_pp.coefs(:,1)=[0.00283938937589918
    5.40181040798507e-006
    2.03529184454081e-007
    1.33082029182343e-007
    1.60929688703906e-008
    1.35082404124139e-008
    2.67713523087757e-009
    2.59288338559281e-009
    1.12645363114093e-010];
S0_pp.coefs(:,2)=[-0.858893627649924
     -0.00707681488016913
    -0.000594642390587056
    -0.000289348613905951
    -8.9725570132443e-005
   -6.55861168268561e-005
   -2.50613955896151e-005
   -1.70299898969802e-005
    -9.251339740199e-006];
S0_pp.coefs(:,3)=[90.6378774936673
         4.04083324065806
        0.972250332355587
        0.530254830109099
        0.340717738089908
        0.263061894610258
        0.172414382193788
        0.130322996707188
        0.104041667070005];
S0_pp.coefs(:,4)=[0
         3314.24084876668
         4143.99963031389
         4506.90534690167
         4716.33086212753
         4866.26995974817
         5077.25397794398
         5227.28409977904
```

```
        5343.16998997484];
elseif strcmpi(fluidtype,'CO2');
S0_pp.coefs(:,1)=[0.00348237135099007
    5.19469384086673e-006
    2.88512295280341e-007
    1.78234000197562e-007
     5.3480428858403e-008
    2.75090625172642e-008
    8.39820025706481e-009
     5.6144152469153e-009
     1.9032303889362e-009];
S0_pp.coefs(:,2)=[-1.05187574122065
     -0.00716433592362781
    -0.000930703314587747
    -0.000497934871667227
    -0.000230583871370884
    -0.000150363228083283
   -6.78360405314897e-005
   -4.26414397602943e-005
   -2.57981940195477e-005];
S0_pp.coefs(:,3)=[111.099260932629
         5.19525321820124
         1.95723752291503
         1.24291842978754
        0.878659058268482
        0.688185508541401
        0.469986239926629
        0.359508759634843
          0.291069125855];
S0_pp.coefs(:,4)=[                    0
         4073.54003204647
         5337.80797736198
         6119.81494708261
         6639.06969408426
         7027.43830898309
         7592.76965195847
         8003.31805161067
         8325.79978673214];
elseif strcmpi(fluidtype,'N2');
S0_pp.coefs(:,1)=[0.00476756426363939
    1.09803222755095e-005
    3.32401093977979e-007
    2.32031935274994e-007
    3.11648974413554e-008
    2.83500841216469e-008
    6.85986614938974e-009
    5.89420191844647e-009
     8.9804892608844e-010];
S0_pp.coefs(:,2)=[-1.44448598883726
      -0.0142167097454451
      -0.0010403230148337
    -0.000541721373866727
    -0.000193673470954222
     -0.00014692612479219
   -6.18758724272538e-005
   -4.12962739790874e-005
   -2.36136682237411e-005];
S0_pp.coefs(:,3)=[ 153.949611278738
         8.07934142046746
         1.97652831635594
         1.18550612200572
        0.817808699595229
        0.647508901722025
        0.438706904502589
        0.335534758096253
        0.270624815893414];
S0_pp.coefs(:,4)= [0
         5717.66550314059
```

```
        7377.46913768897
        8147.20267890576
        8633.52938835131
        8997.91098259053
        9526.84384364201
        9910.53474186674
        10210.6674279024];
elseif strcmpi(fluidtype,'He');
S0_pp.coefs(:,1)=[       0.0211828025756756
    5.35643995955314e-005
    2.07048677446397e-006
    1.36481835840593e-006
    1.30570089689302e-007
    1.38984183639296e-007
    2.51291660107096e-008
    2.69436523468121e-008
    -4.206691];
S0_pp.coefs(:,2)=[         -6.42513085801957
      -0.0702900853168898
       -0.0060128058022521
      -0.0029070756405562 4
     -0.000859848102947319
      -0.00066399296841336
     -0.000247040417495464
     -0.000171652919463337
   -9.08219624229167e-005];
S0_pp.coefs(:,3)=[          689.799279192033
        40.2571848583877
        9.73602841073088
         5.2760876893268
        3.39262581757498
        2.63070528189464
         1.7196718959858
          1.300978559027
        1.03850367714076];
S0_pp.coefs(:,4)=[                       0
        25911.4219146833
          34196.00378145
        37819.6273830604
        39901.5046123855
        41399.1767566473
        43504.8732537679
        45002.6338982689
        46158.9031901794];
elseif strcmpi(fluidtype,'EarthAir');
S0_pp.breaks=[0 100 500 1000 1500 2000 3000]
S0_pp.pieces=6;
S0_pp.order=4;
S0_pp.dim=1;
S0_pp.coefs(:,1)=[       0.0414394391355335
    1.09913650898902e-005
    2.79250504586173e-007
    2.47285910454356e-007
    5.74355887654054e-008
    1.13455667339807e-007
    2.70403090412234e-007];
S0_pp.coefs(:,2)=[          -12.4459809061517
      -0.0141491654916217
     -0.000959527383753384
     -0.000540651626874114
     -0.000169722761192574
    -8.35693780444491e-005
     8.66141229652566e-005];
S0_pp.coefs(:,3)=[          1253.96538745124
        7.95238028691227
        1.90890313676224
        1.15881363144848
       0.803626437415129
```

```
        0.676980367796601
        0.678502740257016];
S0_pp.coefs(:,4)=[          -36612.3489703295
         5763.81984881136
         7384.35285066977
         8133.82888618582
         8608.98353399832
         8975.54551100342
         9307.32530880808];
end
S0=fnval(S0_pp,T);
```

C.10.6 Specific Heat M-file Subroutine Code

Specific heat m-file follows.

```
function Cp=SpecificHeat(fluidtype,T)
% Returns specific heat (joules/kg/Kelvin) for temperatures T for recognized fluid types.
% Results accurate for 100 Kelvins<T<6000 Kelvins, & extrapolated beyond that.
%  Here, T is any array of temperatures in Kelvins, Cp is the resulting array of same size.
%  Recognized fluids include (gasses) 'MarsMix', 'Ar', 'N2','CO2','He','EarthAir', and 'Li' (liquid).
Cp_pp.form='pp';
Cp_pp.breaks=[0 100 500 1000 1500 2000 3000 4000 5000 6000];
Cp_pp.pieces=9;
Cp_pp.order=4;
Cp_pp.dim=1;
if strcmpi(fluidtype,'MarsMix');
Cp_pp.coefs(:,1)=[ 0.000581350983630857
    -9.65450015550039e-007
     3.98026528343767e-007
      3.3614634225973e-008
     5.11730668098727e-008
     5.88396970721698e-009
     1.61139384960509e-009
      1.4224809375556e-009
    -1.59296836722557e-010];
Cp_pp.coefs(:,2)=[-0.173995861194196
      0.000409433895060723
     -0.000749106123599324
     -0.000152066331083674
     -0.000101644379744716
    -2.48847795299067e-005
    -7.23287040825545e-006
    -2.39868885944164e-006
     1.86875395322404e-006];
Cp_pp.coefs(:,3)=[ 18.1956493279625
         0.837006598048948
         0.701137706633507
         0.250551479292009
         0.123696123877816
        0.0604315442405037
        0.0283138943023409
         0.018682335034646
        0.0181524001284304];
Cp_pp.coefs(:,4)=[    0
          660.957304485144
          999.480565919237
          1212.52620437913
          1303.98719053246
          1346.82079088643
```

```
        1388.25152530424
        1410.94394304793
        1428.65007016069];
elseif strcmpi(fluidtype,'Ar');
Cp_pp.coefs(:,1)=[ 0.000520326424351657
    1.08335836872065e-020
   -4.01693493283043e-021
    3.06954461848363e-021
   -3.43586887336439e-022
    -9.473903143468e-024
    1.04212934578148e-022
   -1.13686837721616e-022
    1.13686837721616e-022];
Cp_pp.coefs(:,2)=[ -0.156097927305497
   -9.0002079862946e-018
    3.41060513164848e-018
   -3.41060513164848e-018
    4.54747350886464e-019
   -1.70530256582424e-019
   -1.70530256582424e-019
    1.70530256582424e-019
                       0];
Cp_pp.coefs(:,3)=[ 15.6097927305497
    1.70530256582424e-015
                       0
    6.82121026329696e-016
                       0
    3.41060513164848e-016
                       0
                       0
                       0];
Cp_pp.coefs(:,4)=[ 0
        520.326424351657
        520.326424351657
        520.326424351657
        520.326424351657
        520.326424351657
        520.326424351657
        520.326424351657
        520.326424351657];
elseif strcmpi(fluidtype,'CO2');
Cp_pp.coefs(:,1)=[       0.000575314506242188
   -1.01092237647369e-006
    4.21180499206655e-007
    3.39193603544049e-008
    5.31776873392503e-008
    6.00664609673335e-009
     1.6691240085583e-009
    1.46697941890937e-009
   -1.78241415538877e-010];
Cp_pp.coefs(:,2)=[        -0.172169142761837
     0.000425209110819772
    -0.00078789774094866
   -0.000156126992138678
   -0.000105247951607073
   -2.54814205981964e-005
   -7.46148230799611e-006
   -2.45411028232274e-006
    1.94682797440418e-006];
Cp_pp.coefs(:,3)=[          18.0496670920768
        0.87527372697509
       0.730198274923534
       0.258185908379866
       0.127498436506993
      0.0621337504043581
      0.0291908474981649
      0.0192752549078484
      0.0187679725999319];
```

```
Cp_pp.coefs(:,4)=[                    0
          658.5897878315
         1012.03370425838
         1232.80596888382
         1327.10709508338
         1371.19153635252
         1413.85051225541
         1437.24900145414
         1455.53712549857];
elseif strcmpi(fluidtype,'N2');
Cp_pp.coefs(:,1)=[       0.00104538673122423
    1.25168670955324e-007
   -3.15200587312593e-007
    7.21128979254592e-008
    1.7626251638138e-008
    8.47558704019771e-009
    8.77635838424832e-010
    1.16135210370834e-009
    7.05012371759039e-010];
Cp_pp.coefs(:,2)=[       -0.313459220866882
     0.000156798500387563
     0.000307000905533953
    -0.000165799975434938
    -5.7630628546749e-005
   -3.11912510895422e-005
   -5.76448996894828e-006
    -3.1315824536739e-006
     3.5247385745197e-007];
Cp_pp.coefs(:,3)=[         31.2908448200112
      -0.0393974166383163
        0.146122345730289
        0.216722810779798
        0.105007508788954
       0.0605965689708082
       0.0236408279123164
       0.0147447554896942
       0.0119656468934709];
Cp_pp.coefs(:,4)=[                    0
         1039.87900455653
         1057.21859290435
         1167.62991873891
         1243.55544251076
         1283.85482122331
         1321.73572614478
         1340.48969992657
          1353.2642250663];
elseif strcmpi(fluidtype,'He');
Cp_pp.coefs(:,1)=[       0.0051931244690951
   -2.19353439977862e-019
    5.45696821063757e-020
   -2.72848410531878e-020
    2.58700715170966e-020
    -5.6843418860808e-021
    1.36424205265939e-021
    2.27373675443232e-022
   -1.81898940354586e-021];
Cp_pp.coefs(:,2)=[       -1.55793734072853
    2.12215430413683e-016
   -5.45696821063757e-017
    2.18278728425503e-017
   -2.91038304567337e-017
    1.22781784739345e-017
   -2.72848410531878e-018
                        0
    2.72848410531878e-018];
Cp_pp.coefs(:,3)=[         155.793734072853
   -5.45696821063757e-014
    1.09139364212751e-014
```

```
                    0
    5.45696821063757e-015
   -5.45696821063757e-015
                    0
                    0
                    0];
Cp_pp.coefs(:,4)=[                    0
         5193.12446909509
         5193.12446909509
         5193.12446909509
         5193.12446909509
         5193.12446909509
         5193.12446909509
         5193.12446909509
         5193.12446909509];
elseif strcmpi(fluidtype,'EarthAir');
Cp_pp.breaks=[0 100 500 1000 1500 2000 2500 3000];
Cp_pp.pieces=7;
Cp_pp.coefs(:,1)=[      -0.0219433024220399
   -1.02799909248604e-006
   -1.28735929876774e-007
    5.14388935689814e-009
    4.19542473208218e-007
    5.94054781577479e-007
    1.12868777500029e-006];
Cp_pp.coefs(:,2)=[         6.58432566100875
      0.00133493439678952
     0.000101335485806274
   -9.17684090088861e-005
   -8.40525749735412e-005
     0.000545261134838786
       0.001436343307205];
Cp_pp.coefs(:,3)=[        -658.940534692342
       -0.374475151788105
        0.200032801250212
        0.204816339648904
        0.116905847657693
        0.347510127590315
         1.33831234861221];
Cp_pp.coefs(:,4)=[         23027.7957894584
         1033.69650827185
         1031.70400912383
         1140.96228996591
         1221.07134370775
         1310.95393294424
         1695.28112814628];
elseif strcmpi(fluidtype,'Li');
Cp_pp.breaks=[0 500 1000 1500 2000 ];
Cp_pp.pieces=4;
Cp_pp.order=4;
Cp_pp.coefs(:,1)=[    -3.48687295322885e-006
    6.94329072917594e-009
   -2.63017784945918e-010
    3.60799763081086e-008];
Cp_pp.coefs(:,2)=[       0.00551209812493144
     0.000281788695088167
     0.000292203631181927
     0.000291809104504517  ];
Cp_pp.coefs(:,3)=[         -3.52802041889946
       -0.631077008889664
       -0.344080845754615
      -0.0520744779113993  ];
Cp_pp.coefs(:,4)=[           5184.87959181697
         4363.03479444649
         4118.81137511485
         4019.78898280991 ];
end
Cp=fnval(Cp_pp,T);
```

```
if strcmpi(fluidtype,'Li');
Tmp=453.69;
Cp(find(T<=Tmp))=NaN;
end
```

C.10.7 Thermal Conductivity M-file Subroutine Code

The following m-file returns thermal conductivity.

```
function kth=ThermalConductivity(fluidtype,T)
% Returns thermal conductivity (Watts/meter/Kelvin) for temperatures T for recognized fluid types.
% Results accurate for 100 Kelvins<T<2000 Kelvins, & extrapolated beyond that.
%  Here, T is any array of temperatures in Kelvins, kth is the resulting array of same size.
%  Recognized fluids include (gasses) 'MarsMix', 'Ar', 'N2','CO2','He','EarthAir', and 'Li' (liquid).
kth_pp.form='pp';
kth_pp.breaks=[0 500 1000 1500 2000];
kth_pp.pieces=4;
kth_pp.order=4;
kth_pp.dim=1;
if strcmpi(fluidtype,'MarsMix');
kth_pp.coefs(:,1)=[     -8.43946965540552e-011
    8.68829314643623e-012
   -1.63399992488335e-013
    1.20614460800434e-012
];
kth_pp.coefs(:,2)=[      1.05356931304181e-007
   -2.12351135269022e-008
   -8.20267380724793e-009
   -8.44777379598044e-009];

kth_pp.coefs(:,3)=[ 3.43346633193316e-005
    7.63955722079707e-005
    6.16766785408957e-005
    5.33514547392816e-005];

kth_pp.coefs(:,4)=[ -0.000563169020844222
       0.0323940583956098
       0.0663691027611741
       0.0951363485807489 ];

elseif strcmpi(fluidtype,'Ar');
kth_pp.coefs(:,1)=[      2.58959511719021e-011
    6.59376178597951e-012
    2.10850059204909e-012
    6.33292311079228e-013];
kth_pp.coefs(:,2)=[  -5.48845783903993e-008
   -1.60406516325461e-008
   -6.15000895357691e-009
   -2.98725806550341e-009];
kth_pp.coefs(:,3)=[      7.55998625450829e-005
    4.01372475336103e-005
    2.90419172405488e-005
    2.44732837310088e-005];
kth_pp.coefs(:,4)=[ -0.000640792446023078
       0.0266749881254063
       0.0435576692073224
       0.0568046881632087];
elseif strcmpi(fluidtype,'CO2');
kth_pp.coefs(:,1)=[ -8.95697919772388e-011
     8.7159435478886e-012
   -2.53405571179377e-013
    1.21190485215506e-012];
```

```
kth_pp.coefs(:,2)=[ 1.1296715060041e-007
   -2.13875373654485e-008
   -8.31362204361567e-009
   -8.69373040038474e-009];
kth_pp.coefs(:,3)=[ 3.19226886475612e-005
    7.77124952650418e-005
    6.28619155605098e-005
    5.43582393385096e-005];
kth_pp.coefs(:,4)=[-0.000558147134909062
       0.0324487608418191
        0.067047617076464
       0.0963684936494176];
elseif strcmpi(fluidtype,'N2');
kth_pp.coefs(:,1)=[3.90252304075642e-011
    1.05689751803369e-011
    1.38622644656266e-012
    1.82829825580045e-012];
kth_pp.coefs(:,2)=[-7.94217627419242e-008
   -2.08839171305778e-008
   -5.03045436007238e-009
   -2.95111469022847e-009];
kth_pp.coefs(:,3)=[0.000108965874681138
    5.88130347448874e-005
    4.58558489995622e-005
    4.18650644744119e-005];
kth_pp.coefs(:,4)=[ -0.000732551043351641
       0.0387730994126821
       0.0642797594000234
       0.0861233486156067];
elseif strcmpi(fluidtype,'He');
kth_pp.coefs(:,1)=[ 3.92617328131614e-010
    2.98637889870604e-011
   -6.69940393922385e-012
    1.8145560550439e-011];

kth_pp.coefs(:,2)=[-6.27680083304969e-007
   -3.87540911075476e-008
    6.04159237304301e-009
   -4.00751353579309e-009];
kth_pp.coefs(:,3)=[ 0.00058211017104374
     0.000248893083837481
     0.000232536834470229
     0.000233553873888854];
kth_pp.coefs(:,4)=[ 0.0217316847044791
        0.204943915416558
        0.323434908181795
        0.440376298017767];
elseif strcmpi(fluidtype,'EarthAir');
kth_pp.coefs(:,1)=[     4.33404489204219e-011
    1.05934327817733e-011
   -4.64891217598021e-013
    5.75818216393941e-012];
kth_pp.coefs(:,2)=[     -8.5179120594456e-008
   -2.01684472138231e-008
   -4.27829804116303e-009
   -4.97563486756013e-009];
kth_pp.coefs(:,3)=[      0.000116367375504442
    6.36935916003023e-005
    5.14702189728091e-005
    4.68432525184476e-005];
kth_pp.coefs(:,4)=[      -0.00106933943641563
        0.041237124282244
       0.0693659873766611
       0.0939734109505751];
elseif strcmpi(fluidtype,'Li');
kth_pp.breaks=[0 500 1000 1500];
lth_pp.pieces=3;
kth_pp.coefs(:,1)=[  -3.96780228307425e-009
```

```
     -1.373130078214e-008
   -1.65331460247586e-009 ];
kth_pp.coefs(:,2)=[       1.03284191734894e-005
    4.37671574887792e-006
   -1.62202354243321e-005 ];
kth_pp.coefs(:,3)=[          0.0252929686003196
       0.0326455360615033
       0.0267237762237763];
kth_pp.coefs(:,4)=[           29.5690983740431
         44.301712182191
        60.0022465523946];
end
kth=fnval(kth_pp,T);
if strcmpi(fluidtype,'Li');
Tmp=453.69;
kth(find(T<=Tmp))=NaN;
end
```

C.10.8 Viscosity M-file Subroutine Code

The following m-file code returns fluid viscosity.

```
function mu=Viscosity(fluidtype,T)
% Returns viscosity (Pascal-Seconds) for temperatures T for recognized fluid types.
% Results accurate for 100 Kelvins<T<2000 Kelvins, & extrapolated beyond that.
%  Here, T is any array of temperatures in Kelvins, mu is the resulting array of same size.
%  Recognized fluids include (gasses) 'MarsMix', 'Ar', 'N2','CO2','He','EarthAir', and 'Li' (liquid).
mu_pp.form='pp';
mu_pp.breaks=[0 500 1000 1500 2000];
mu_pp.pieces=4;
mu_pp.order=4;
mu_pp.dim=1;
if strcmpi(fluidtype,'MarsMix');
mu_pp.coefs(:,1)=[ 1.91259324527568e-014
    7.06525325429854e-015
    1.59489983582924e-015
   -1.36251102956505e-015];
mu_pp.coefs(:,2)=[          -4.4580839763486e-011
   -1.58919410843509e-011
   -5.29406120290307e-012
   -2.90171144915931e-012];
mu_pp.coefs(:,3)=[      6.80145875596715e-008
    3.77781971357531e-008
     2.7185195992126e-008
    2.30873096660949e-008];
mu_pp.coefs(:,4)=[ -1.68253261087149e-006
    2.35702927846874e-005
    3.93695627382635e-005
    5.18380079130794e-005 ];
elseif strcmpi(fluidtype,'Ar');
mu_pp.coefs(:,1)=[ 3.40044524999721e-014
    8.58057752977249e-015
    2.81766298833804e-015
   -1.71211390699328e-015];
mu_pp.coefs(:,2)=[-7.1216312951265e-011
   -2.02096342013069e-011
   -7.33876790664809e-012
   -3.11227342414123e-012];
mu_pp.coefs(:,3)=[ 9.66890140720963e-008
    5.09760404958105e-008
```

```
    3.72018394418329e-008
    3.19763187764384e-008];
mu_pp.coefs(:,4)=[-7.64588468456918e-007
    3.40263968922715e-005
    5.55345807810716e-005
    7.26530163988682e-005];
elseif strcmpi(fluidtype,'CO2');
mu_pp.coefs(:,1)=[ 1.84521269180779e-014
    7.03264658699861e-015
     1.5547749929767e-015
   -1.36947452041497e-015];
mu_pp.coefs(:,2)=[ -4.34734084418281e-011
   -1.57952180647113e-011
   -5.24624818421333e-012
   -2.91408569474838e-012];
mu_pp.coefs(:,3)=[6.70617473216595e-008
    3.74274340683898e-008
    2.69067009439274e-008
    2.28265340044466e-008];
mu_pp.coefs(:,4)=[-1.73497359299849e-006
    2.32340638221339e-005
    3.88780571635258e-005
    5.12141924635583e-005];
elseif strcmpi(fluidtype,'N2');
mu_pp.coefs(:,1)=[3.43204234160597e-014
    6.45842982731327e-015
    1.93145417908812e-015
   -3.5605237744431e-016];
mu_pp.coefs(:,2)=[    -6.58717146857726e-011
   -1.43910795616831e-011
   -4.70343482071328e-012
   -1.80625355208116e-012];
mu_pp.coefs(:,3)=[    7.66182495560525e-008
    3.64868524323247e-008
    2.69395952411266e-008
    2.36847510547294e-008];
mu_pp.coefs(:,4)=[-9.49371077254447e-008
    2.60363119258651e-005
    4.14892719800209e-005
    5.40246426677918e-005];
elseif strcmpi(fluidtype,'He');
mu_pp.coefs(:,1)=[  1.91695479899686e-014
    6.24463127005045e-015
   -6.0365993852115e-015
    1.70236468954153e-014];
mu_pp.coefs(:,2)=[    -4.17821486162275e-011
   -1.30278266312746e-011
   -3.66087972619893e-012
   -1.27157788040163e-011];
mu_pp.coefs(:,3)=[    6.76064817179773e-008
    4.02014940942262e-008
    3.18571409154894e-008
    2.36688116503818e-008];
mu_pp.coefs(:,4)=[    2.53036583099191e-006
    2.82842630346697e-005
    4.59086323327205e-005
     6.0167407935764e-005];
elseif strcmpi(fluidtype,'EarthAir');
mu_pp.form='pp';
mu_pp.breaks=[0 500 1000 1500 2000 2500 3000];
mu_pp.pieces=6;
mu_pp.order=4;
mu_pp.dim=1;
mu_pp.coefs(:,1)=[    4.62055101087152e-014
    2.44033392827618e-015
    7.07218511033621e-015
    1.87071169061818e-016
   -8.13805121176328e-015
```

```
    1.97452839499109e-014];
mu_pp.coefs(:,2)=[    -8.16494844583416e-011
   -1.23412192952689e-011
   -8.68071840285467e-012
    1.92755926264958e-012
    2.20816601624222e-012
   -9.99891080140277e-012];
mu_pp.coefs(:,3)=[     8.43542817634652e-008
     3.735892988666e-008
    2.68479610375982e-008
    2.34713814674957e-008
    2.55392441069417e-008
    2.16438717143615e-008];
mu_pp.coefs(:,4)=[    -6.56284588420321e-007
    2.68841739423163e-005
    4.27833758028636e-005
     5.4921199859741e-005
     6.7162164305284e-005
     7.9466571461345e-005];
elseif strcmpi(fluidtype,'Li');
mu_pp.breaks=[0 500 1000 1500 2000 2500 3000 3500]
mu_pp.pieces=7;
mu_pp.coefs(:,1)=[     -1.95331309831702e-010
   -9.80388718870332e-013
   -5.77397340032479e-014
    -3.9437989606833e-014
    6.21108299029389e-016
   -1.02105094854462e-014
    1.03295377449395e-014  ];
mu_pp.coefs(:,2)=[       2.94630688965853e-007
    1.63372421830003e-009
    1.63141139994535e-010
    7.65315389896636e-011
     1.7374554579414e-011
     1.8306217027958e-011
    2.99045279978884e-012 ];
mu_pp.coefs(:,3)=[       -0.000149251629630418
   -1.11942303834102e-006
   -2.20990359193739e-007
    -1.0115401970164e-007
    -5.4200972917101e-008
    -3.6360587113415e-008
   -2.57122521995416e-008 ];
mu_pp.coefs(:,4)=[         0.0259134142185961
      0.00052885791588785
     0.000255028861433555
     0.000178101500084913
     0.000141727626280655
     0.000119048417004336
     0.000104168364018938  ];
end
mu=fnval(mu_pp,T);
if strcmp(fluidtype,'Li');
Tmp=453.69;
mu(find(T<=Tmp))=NaN;
end
```

APPENDIX D
D MATLAB-ASSISTED SHIELD CALCULATIONS BY SCALE

This MSWORD-MATLAB notebook document was used to guide, document, and partially automate a manual sshield optimization search procedure using SCALE5. Fundamental is its calculations of shield mass, which are implemented internally as embedded MATLAB procedures. Embedded MATLAB scripts were also used to create input lines for SCALE5's material information processor from user-input changes to the masses of shield material layers. Communications both from this file to the SCALE input file and from the SCALE output file to this file were implemented through the "cut-and-paste" capability of software running under Windows on a PC. This document also functioned as a journal documenting progress of the manual search, since it records the sequence of SCALE run inputs and their corresponding results and since the user is naturally typinc comments into the document during the process.

When the manual search had been completed for a single case, this document;s length had grown to 206 pages. However, it has been decided to abridge this document for inclusion here since details of the manual search have been superseded by the development of more automated ways to optimize a shield design.

The manual iterative procedure followed

We assume the shield is divided into N zones, and that their limiting radii are stored in vector r(N+1). Each zone is composed of a mix of M=4 materials. Note that the materials are as follows:
(1) Tungsten
(2) Boron Carbide using the B-10 isotope
(3) Solid Lithium Hydride using the Li-6 isotope
(4) Lead

The densities of the materials are given by the vector, rhomat(M). The volume fractions with each zone are given in the matrix, u(N,M). Clearly it is necessary that the sum of u within each zone must be 1.00.

The total mass is then calculated as follows.

```
rhomat=[19.3748; 2.52; 0.700; 11.344],
r=[8.7731; 28.77; 78.17; 80; 136]
u=[1 0 0 0;0.45 0.05 0.5 0.; 0 0 0 1; 0 0 1 0],
[N,M]=size(u),
Mass=4*pi/3*(r(2:N+1,1).^3-r(1:N,1).^3)'*u*rhomat/1e6
```

```
rhomat =
   19.3748
    2.5200
    0.7000
```

```
   11.3440
r =
    8.7731
   28.7700
   78.1700
   80.0000
  136.0000
u =
    1.0000         0         0         0
    0.4500    0.0500    0.5000         0
         0         0         0    1.0000
         0         0    1.0000         0
N =
     4
M =
     4
Mass =
   26.8637
```

Final calculated radiation dose output @ 6m is as follows:
neutrons=2.112E-21
gammas=8.752E-21
total= 1.0864E-20

Next, changed outer radius to 137 cm.

Final calculated radiation dose output @ 6m is as follows:
neutrons=1.849E-21
gammas=8.547E-21
total= 1.0396E-20

Next, changed radius to 138.85 cm
neutrons=1.444E-21
gammas=8.178E-21
total= 0.9622E-20

Next changed radius to 137.85
neutrons=1.650E-21
gammas=8.376E-21
total= 1.0026E-20

Next, tried 137.91
neutrons=1.637E-21
gammas=8.364E-21
total= 1.0001E-20

GOOD ENOUGH FOR BASELINE !

```
rhomat=[19.3748; 2.52; 0.700; 11.344],
r=[8.7731; 28.77; 78.17; 80; 137.91]
u=[1 0 0 0;0.45 0.05 0.5 0.; 0 0 0 1; 0 0 1 0],
[N,M]=size(u),
Mass=4*pi/3*(r(2:N+1,1).^3-r(1:N,1).^3)'*u*rhomat/1e6

rhomat =
   19.3748
    2.5200
    0.7000
   11.3440
r =
    8.7731
   28.7700
   78.1700
   80.0000
  137.9100
u =
    1.0000         0         0         0
    0.4500    0.0500    0.5000         0
         0         0         0    1.0000
         0         0    1.0000         0
N =
     4
M =
     4
Mass =
   27.1788
```

Next, we perturb the tunsten content in the second zone from 0.45 to 0.46 while reducing its LiH ontent from 0.50 to 0.49, and balance the changed attenuation by changing the outer radius of the LiH.
Initial result with outer radius still at 137.91 cm is
neutrons=1.566E-21
gammas=8.046E-21
total=0.9612E-20

Need to reduce shielding. Drop outer radius down to 137.00 cm
result is
neutrons=1.768E-21
gammas=8.222E-21
total=0.9990E-20

Set radius to 136.976
neutrons=1.774E-21
gammas=8.227E-21

total=1.0001E-20
GOOD ENOUGH !

```
rhomat=[19.3748; 2.52; 0.700; 11.344],
r=[8.7731; 28.77; 78.17; 80; 136.976]
u=[1 0 0 0;0.46 0.05 0.49 0.; 0 0 0 1; 0 0 1 0],
[N,M]=size(u),
Mass=4*pi/3*(r(2:N+1,1).^3-r(1:N,1).^3)'*u*rhomat/1e6

rhomat =
   19.3748
    2.5200
    0.7000
   11.3440
r =
    8.7731
   28.7700
   78.1700
   80.0000
  136.9760
u =
    1.0000         0         0         0
    0.4600    0.0500    0.4900         0
         0         0         0    1.0000
         0         0    1.0000         0
N =
     4
M =
     4
Mass =
   27.3786
```

So the initial perturbation took the mass at constant attenuation from 27.1788 tonnes to 27.3765 tonnes. This is the wrong direction. So we need to consider the reverse direction. Next, we drop the tungsten content to 0.40 and increase the LiH content to 0.55.
Initial result with outer radius still at 136.976 cm is
neutrons=2.316E-21
gammas=1.116E-20
total=1.3476E-20

Set outer radius to 138 cm
neutrons=2.020E-21
gammas=1.090E-20
total=1.292E-20

Set outer radius to 143 cm
neutrons=1.038E-21
gammas=9.678E-21
total=1.0716E-20

Set outer radius to 144 cm
neutrons=9.086E-22
gammas=9.450E-21
total=1.03586E-20

Set outer radius to 145 cm
neutrons=7.953E-22
gammas=9.226E-21
total=1.00213E-20

Set outer radius to 145.1 cm
neutrons=7.848E-22
gammas=9.204E-21
total=0.99888E-20

Set outer radius to 145.066
neutrons=7.884E-22
gammas=9.212E-21
total=1.00004E-20
GOOD ENOUGH !

```
rhomat=[19.3748; 2.52; 0.700; 11.344],
r=[8.7731; 28.77; 78.17; 80; 145.066]
u=[1 0 0 0;0.40 0.05 0.55 0.; 0 0 0 1; 0 0 1 0],
[N,M]=size(u),
Mass=4*pi/3*(r(2:N+1,1).^3-r(1:N,1).^3)'*u*rhomat/1e6

rhomat =
   19.3748
    2.5200
    0.7000
   11.3440
r =
    8.7731
   28.7700
   78.1700
   80.0000
  145.0660
u =
    1.0000         0         0         0
    0.4000    0.0500    0.5500         0
         0         0         0    1.0000
         0         0    1.0000         0
N =
     4
M =
     4
Mass =
   26.6641
```

Note that this indeed was a mass reduction, albeit a small one.

Since the neutron and gamma signals are so unbalanced, it may be that this last step in tungsten content may have been too large. Next I shall cut the step in half to see what that result would be. So, I shall set the tungsten VF to 0.425 and the LiH volume fraction to 0.525.
Initial result, with the same outer radius:
neutrons=7.052E-22
gammas=7.916E-21
total=0.86212E-20
Need to reduce shielding.

Set outer radius below 145.066 cm, to 140 cm
neutrons=1.385E-21
gammas=8.937E-21
total=1.0322E-20

Set outer radius to 141 cm neutrons=1.212E-21
gammas=8.727E-21
total=0.9939E-20

Set outer radius to 140.841 cm
neutrons=1.238E-21
gammas=8.760E-21
total=0.9998E-20

Set outer radius to 140.81
neutrons=1.243E-21
gammas=8.766E-21
total=1.0009

Set outer radius to 140.835
neutrons=1.239E-21
gammas=8.761E-21
total=1.0000E-20
GOOD ENOUGH!

```
rhomat=[19.3748; 2.52; 0.700; 11.344],
r=[8.7731; 28.77; 78.17; 80; 140.835]
u=[1 0 0 0;0.425 0.05 0.525 0.; 0 0 0 1; 0 0 1 0],
[N,M]=size(u),
Mass=4*pi/3*(r(2:N+1,1).^3-r(1:N,1).^3)'*u*rhomat/1e6

rhomat =
```

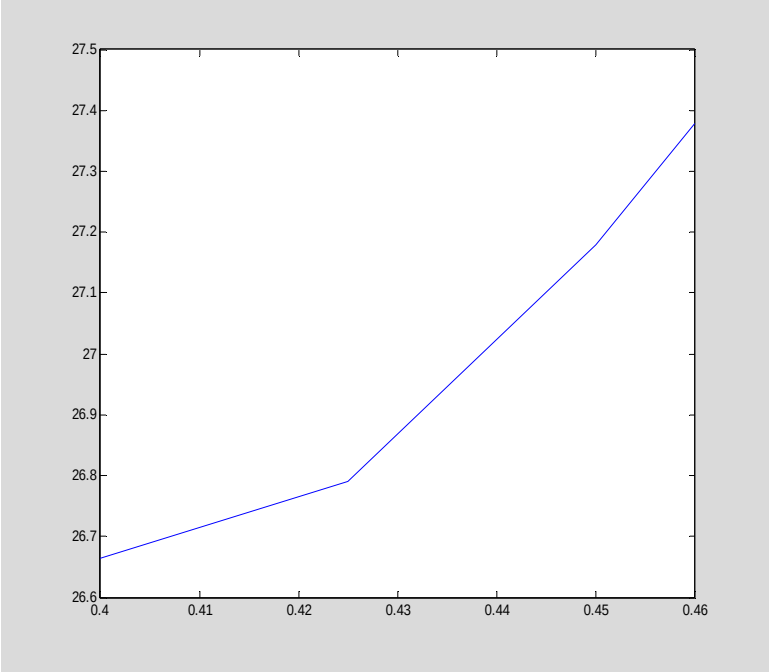
```
   19.3748
    2.5200
    0.7000
   11.3440
r =
    8.7731
   28.7700
   78.1700
   80.0000
  140.8350
u =
    1.0000         0         0         0
    0.4250    0.0500    0.5250         0
         0         0         0    1.0000
         0         0    1.0000         0
N =
     4
M =
     4
Mass =
   26.7911
```

Next, I collect the results so far of this scan.

Volume Fraction of Tungsten in Zone 2	Resulting Mass with radiation leakage=1E-20
0.40	26.6641
0.425	26.7911
0.45	27.1788
0.46	27.3786

A plot of this data follows

```
plot([.4 .425 .45 .46],[26.6641 26.7911   27.1788 27.3786])
```



Next we try 0.375 for the tungsten volume fraction.
We leave the outer radius as 140.835 cm
neutrons=1.548E-21
gammas=1.244E-20
total=1.3988E-20

Increase the outer radius to 145 cm
neutrons=8.891E-22
gammas=1.127E-20
total=1.21951E-20

Increase router to 150 cm
neutrons=4.571E-22
gammas=1.000E-20
total=1.04571E-20

Increase outer radius to 151 cm
neutrons=4.001E-22
gammas=9.763E-21
total=1.01631E-20

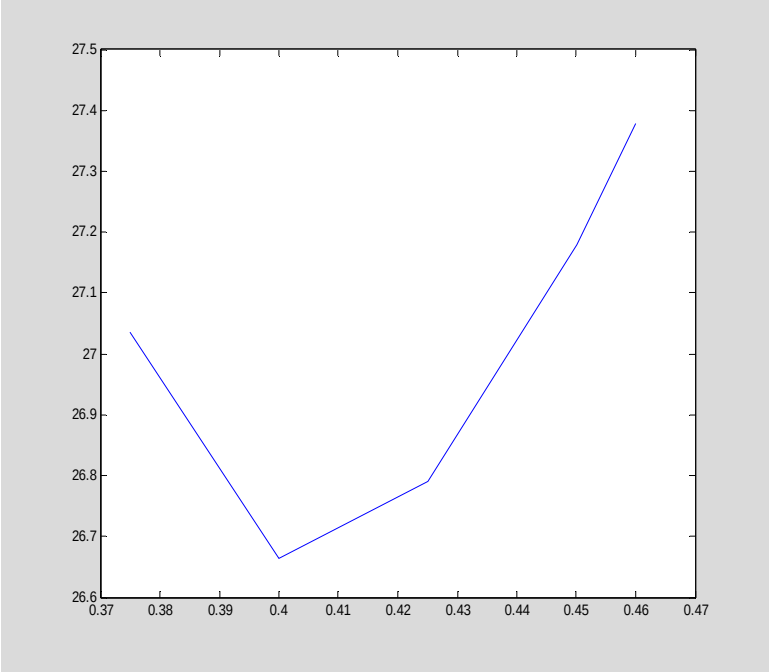
Increase to 151.5
neutrons=3.744E-22
gammas=9.647E-21
total=1.00214E-20

Increase to 151.575
neutrons=3.707E-22
gammas=9.629E-21
total=0.99997E-20
GOOD ENOUGH

```
rhomat=[19.3748; 2.52; 0.700; 11.344],
r=[8.7731; 28.77; 78.17; 80; 151.575]
u=[1 0 0 0;0.375 0.05 0.575 0.; 0 0 0 1; 0 0 1 0],
[N,M]=size(u),
Mass=4*pi/3*(r(2:N+1,1).^3-r(1:N,1).^3)'*u*rhomat/1e6

rhomat =
   19.3748
    2.5200
    0.7000
   11.3440
r =
    8.7731
   28.7700
   78.1700
   80.0000
  151.5750
u =
    1.0000         0         0         0
    0.3750    0.0500    0.5750         0
         0         0         0    1.0000
         0         0    1.0000         0
N =
     4
M =
     4
Mass =
   27.0363
```

Volume Fraction of Tungsten in Zone 2	Resulting Mass with radiation leakage=1E-20
0.375	27.0363
0.40	26.6641
0.425	26.7911
0.45	27.1788
0.46	27.3786

```
plot([.375 .4 .425 .45 .46],[27.0363 26.6641 26.7911 27.1788 27.3786])
```



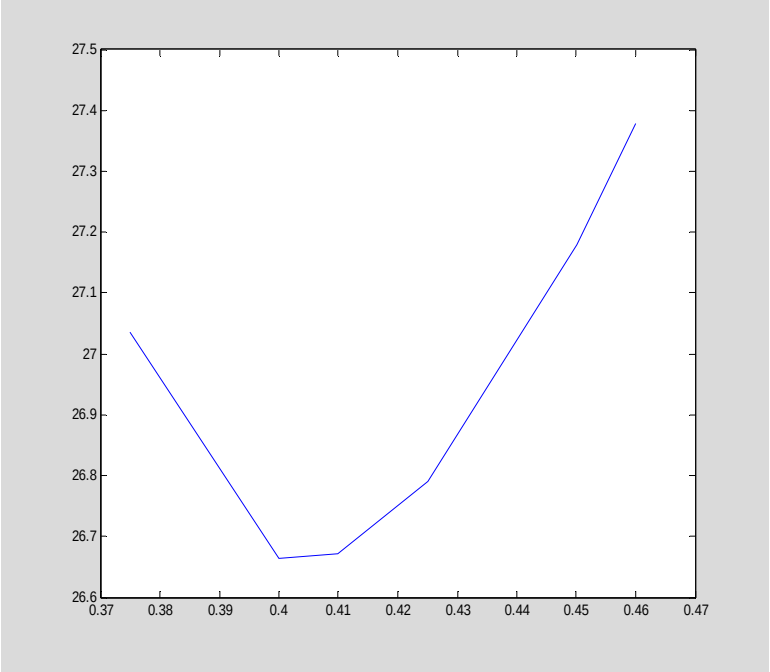
This looks as though there is a minimum perhaps arouund 0.41, which I shall try next.
r_outer=151.575
neutrons=3.170E-22
gammas=7.374E-21
total=7.691E-21
---
r_outer=140
neutrons=1.480E-21
gammas=9.737E-21
total=1.1217E-20
---
r_outer=143 cm
neutrons=9.927E-22
gammas=9.065E-21
total=1.00577E-20
---
r_outer=143.1493
neutrons=9.732E-22
gammas=9.033E-21
total=1.00062E-20
---
r_outer=143.164
neutrons=9.712E-22
gammas=9.029E-21

total=1.00002E-20
GOOD ENOUGH !

```
rhomat=[19.3748; 2.52; 0.700; 11.344],
r=[8.7731; 28.77; 78.17; 80; 143.164]
u=[1 0 0 0;0.41 0.05 0.54 0.; 0 0 0 1; 0 0 1 0],
[N,M]=size(u),
Mass=4*pi/3*(r(2:N+1,1).^3-r(1:N,1).^3)'*u*rhomat/1e6

rhomat =
   19.3748
    2.5200
    0.7000
   11.3440
r =
    8.7731
   28.7700
   78.1700
   80.0000
  143.1640
u =
    1.0000         0         0         0
    0.4100    0.0500    0.5400         0
         0         0         0    1.0000
         0         0    1.0000         0
N =
     4
M =
     4
Mass =
   26.6716
```

Volume Fraction of Tungsten in Zone 2	Resulting Mass with radiation leakage=1E-20
0.375	27.0363
0.40	26.6641
0.41	26.6716
0.425	26.7911
0.45	27.1788
0.46	27.3786

```
plot([.375 .4 .41 .425 .45 .46],[27.0363 26.6641 26.6716 26.7911
27.1788 27.3786])
```



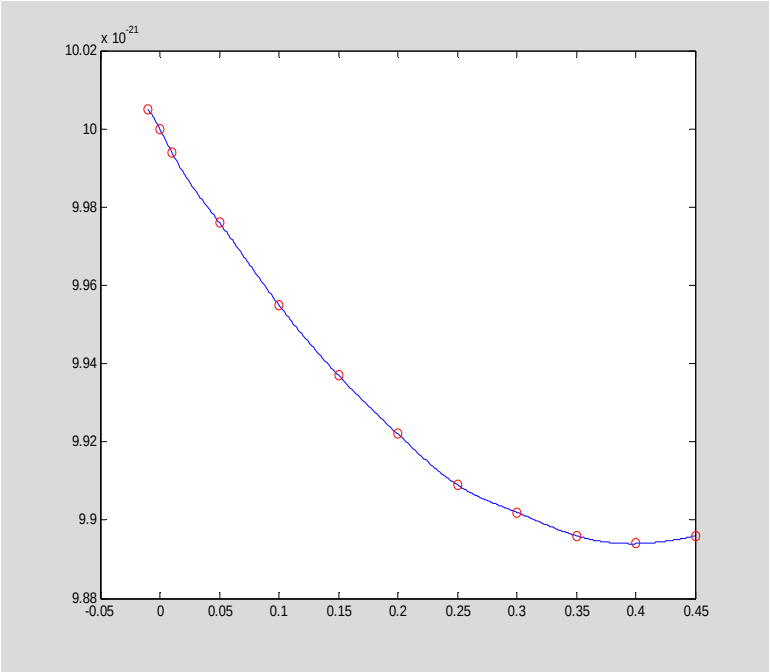
INTENTIONALLY DELETED 190 PAGES

Summary of line search results so far.

Multiplier of dM	neutrons	Gammas	total
+0.01	1.731E-21	8.274E-21	1.0005E-20
0	1.737E-21	8.263E-21	1.0000E-20
-0.01	1.741E-21	8.253E-21	0.9994E-20
-0.05	1.762E-21	8.214E-21	0.9976E-20
-0.10	1.787E-21	8.168E-21	0.9955E-20
-0.15	1.813E-21	8.124E-21	0.9937E-20
-0.20	1.840E-21	8.082E-21	0.9922E-20
-0.25	1.866E-21	8.043E-21	0.9909E-20
-0.30	1.894E-21	8.008E-21	0.9902E-20
-0.35	1.922E-21	7.974E-21	0.9896E-20
-0.40	1.950E-21	7.944E-21	0.9894E-20
-0.45	1.979E-21	7.917E-21	0.9896E-20

```
mult= [-.01;0;.01;.05;0.1;0.15;0.2;0.25;0.3;0.35;0.4;0.45];
Leakage=[1.0005E-20; 1.0000E-20; 0.9994E-20; 0.9976E-20; 0.9955E-20;
0.9937E-20; 0.9922E-20; 0.9909E-20; 0.9902E-20; 0.9896E-20; 0.9894E-20;
0.9896E-20]; XX=linspace(-0.01,0.45,461);
YY=spline(mult,Leakage,XX);plot(XX,YY,mult,Leakage,
'ro'),I=find(YY==min(YY)),XX(I),YY(I)

I =
```

```
   408
ans =
    0.3970
ans =
  9.8940e-021
```



ans =
  22.9105

Thus, the minimum value occurs at mult = -0.40 tonne.
Next, reduce mass to increase leakage to 1E-20.

```
[r,f]=m2rf(Mass+[0;0;0;0;0;0.12;0;-0.13]);1-f,
['W-182  15  DEN=5.1022 ' num2str(f(3),'%019.16f')  ' 873 END';
'W-183  15  DEN=2.7742 ' num2str(f(3),'%019.16f')   ' 873 END';
'W-184  15  DEN=5.9500 ' num2str(f(3),'%019.16f')   ' 873 END';
'W-186  15  DEN=5.5484 ' num2str(f(3),'%019.16f')   ' 873 END';,
'B-10   15  DEN=2.37   ' num2str(1-f(3),'%019.16f') ' 873 END',
'W-182  16  DEN=5.1022 ' num2str(f(4),'%019.16f')   ' 873 END';
'W-183  16  DEN=2.7742 ' num2str(f(4),'%019.16f')   ' 873 END'
'W-184  16  DEN=5.9500 ' num2str(f(4),'%019.16f')   ' 873 END';
'W-186  16  DEN=5.5484 ' num2str(f(4),'%019.16f')   ' 873 END';
'B-10   16  DEN=2.37   ' num2str(1-f(4),'%019.16f') ' 873 END';
'Pb     17  DEN=11.344 ' num2str(  f(5),'%019.16f') ' 873 END';
'B-10   17  DEN=2.37   ' num2str(1-f(5),'%019.16f') ' 873 END'],
[[1;2;15;16;17;10] r]

ans =
```

```
         0
         0
    0.6344
    0.8164
    0.0003
ans =
W-182  15  DEN=5.1022 00.3655706736932543 873 END
W-183  15  DEN=2.7742 00.3655706736932543 873 END
W-184  15  DEN=5.9500 00.3655706736932543 873 END
W-186  15  DEN=5.5484 00.3655706736932543 873 END
B-10   15  DEN=2.37   00.6344293263067458 873 END
W-182  16  DEN=5.1022 00.1836496437525630 873 END
W-183  16  DEN=2.7742 00.1836496437525630 873 END
W-184  16  DEN=5.9500 00.1836496437525630 873 END
W-186  16  DEN=5.5484 00.1836496437525630 873 END
B-10   16  DEN=2.37   00.8163503562474370 873 END
Pb     17  DEN=11.344 00.9997031037539840 873 END
B-10   17  DEN=2.37   00.0002968962460160 873 END
ans =
    1.0000    8.7731
    2.0000   23.9255
   15.0000   54.4823
   16.0000   82.7336
   17.0000   84.3539
   10.0000  135.6522
```

Results:

neutrons=2.249E-21

gammas=7.758E-21

total=1.0007E-20

```
[r,f]=m2rf(Mass+[0;0;0;0;0;0.12225;0;-0.13]);1-f,
['W-182  15  DEN=5.1022 ' num2str(f(3),'%019.16f')  ' 873 END';
'W-183  15  DEN=2.7742 ' num2str(f(3),'%019.16f')   ' 873 END';
'W-184  15  DEN=5.9500 ' num2str(f(3),'%019.16f')   ' 873 END';
'W-186  15  DEN=5.5484 ' num2str(f(3),'%019.16f')   ' 873 END';,
'B-10   15  DEN=2.37   ' num2str(1-f(3),'%019.16f') ' 873 END',
'W-182  16  DEN=5.1022 ' num2str(f(4),'%019.16f')   ' 873 END';
'W-183  16  DEN=2.7742 ' num2str(f(4),'%019.16f')   ' 873 END'
'W-184  16  DEN=5.9500 ' num2str(f(4),'%019.16f')   ' 873 END';
'W-186  16  DEN=5.5484 ' num2str(f(4),'%019.16f')   ' 873 END';
'B-10   16  DEN=2.37   ' num2str(1-f(4),'%019.16f') ' 873 END';
'Pb     17  DEN=11.344 ' num2str(  f(5),'%019.16f') ' 873 END';
'B-10   17  DEN=2.37   ' num2str(1-f(5),'%019.16f') ' 873 END'],
[[1;2;15;16;17;10] r]

ans =
         0
         0
    0.6344
    0.8164
    0.0003
ans =
W-182  15  DEN=5.1022 00.3655706736932543 873 END
```

```
W-183  15  DEN=2.7742 00.3655706736932543 873 END
W-184  15  DEN=5.9500 00.3655706736932543 873 END
W-186  15  DEN=5.5484 00.3655706736932543 873 END
B-10   15  DEN=2.37   00.6344293263067458 873 END
W-182  16  DEN=5.1022 00.1836496437525630 873 END
W-183  16  DEN=2.7742 00.1836496437525630 873 END
W-184  16  DEN=5.9500 00.1836496437525630 873 END
W-186  16  DEN=5.5484 00.1836496437525630 873 END
B-10   16  DEN=2.37   00.8163503562474370 873 END
Pb     17  DEN=11.344 00.9997035175329104 873 END
B-10   17  DEN=2.37   00.0002964824670896 873 END
ans =
    1.0000    8.7731
    2.0000   23.9255
   15.0000   54.4823
   16.0000   82.7336
   17.0000   84.3561
   10.0000  135.6530
```

Results:

neutrons=2.249E-21

gammas=7.751E-21

total=1.0000E-20

GOOD ENOUGH !

```
Mass=Mass+[0;0;0;0;0;0.12225;0;-0.13],sum(Mass)

Mass =
    1.0567
    4.3917
    0.9323
    6.0300
    3.2788
    1.6140
    0.0001
    5.5593
ans =
   22.8629

Mass =
    1.0567
    4.3917
    0.9323
    6.0300
    3.2788
    1.6140
    0.0001
    5.5593
ans =
   22.8629
```

At this point, it is appropriate to declare the optimization essentially finished. The minimum shield mass found is 22,862.9 kilograms (i.e., 22.8629 tonnes). This shield guarantees that with one megawatt continuous fission power in the reactor, the

radiation leakage 6 meters away from the reactor center is so low that the annual dose is approximately the official USA limit for radiation workers.

Since the shield is spherical, leakage is equally attenuated in all directions including directions away from the crewed vehicle such as sideways, backwards, up, and down.

Note that the actual leakage according to the documentation writeup for SAS1 & XDOSE codes is 1.0E-20 rem/hr per fission neutron/second. There appears to be some confusion here since it states this is normalized to 1 fission neutron per fission. It may be necessary to multiply by nu, the average number of neutrons from each fission event, which may be about 2. However, that may not be appropriate for the gamma portion of the radiation dose.

Applying the documentation's formula, the shield leakage at full 1 MW power would be about (nu/3) millirem/hr at a distance of 6 meters from the reactor's center, which would add up to (2.92 nu) rem over the course of one full-power year. The dose at 2 meters from the reactor center is about ten times that value.

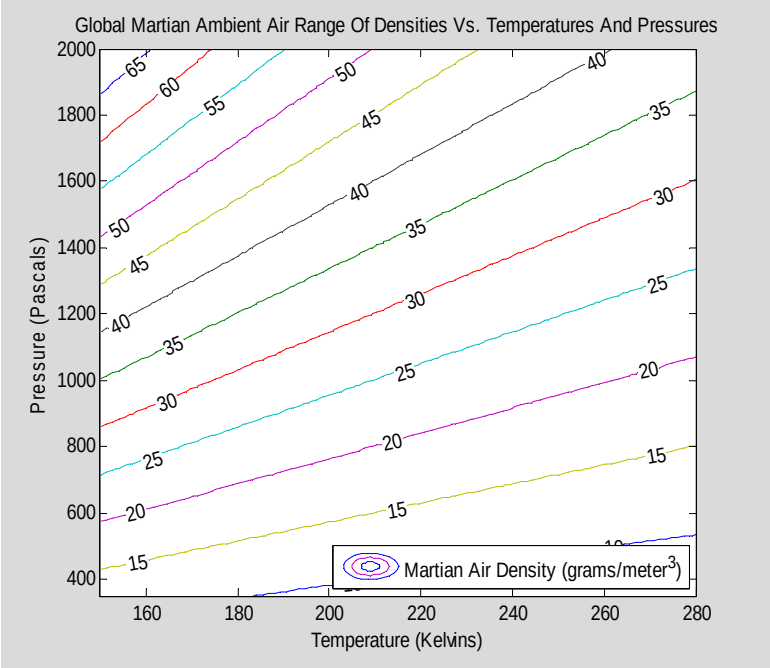
```
[r,f]=m2rf(Mass),1-f,
['W-182  15  DEN=5.1022 ' num2str(f(3),'%019.16f')  ' 873 END';
'W-183  15  DEN=2.7742 ' num2str(f(3),'%019.16f')   ' 873 END';
'W-184  15  DEN=5.9500 ' num2str(f(3),'%019.16f')   ' 873 END';
'W-186  15  DEN=5.5484 ' num2str(f(3),'%019.16f')   ' 873 END';,
'B-10   15  DEN=2.37   ' num2str(1-f(3),'%019.16f') ' 873 END',
'W-182  16  DEN=5.1022 ' num2str(f(4),'%019.16f')   ' 873 END';
'W-183  16  DEN=2.7742 ' num2str(f(4),'%019.16f')   ' 873 END'
'W-184  16  DEN=5.9500 ' num2str(f(4),'%019.16f')   ' 873 END';
'W-186  16  DEN=5.5484 ' num2str(f(4),'%019.16f')   ' 873 END';
'B-10   16  DEN=2.37   ' num2str(1-f(4),'%019.16f') ' 873 END';
'Pb     17  DEN=11.344 ' num2str(  f(5),'%019.16f') ' 873 END';
'B-10   17  DEN=2.37   ' num2str(1-f(5),'%019.16f') ' 873 END'],
[[1;2;15;16;17;10] r]

r =
    8.7731
   23.9255
   54.4823
   82.7336
   84.3561
  135.6530
f =
    1.0000
    1.0000
    0.3656
    0.1836
    0.9997
ans =
         0
```

```
         0
    0.6344
    0.8164
    0.0003
ans =
W-182  15  DEN=5.1022 00.3655706736932543 873 END
W-183  15  DEN=2.7742 00.3655706736932543 873 END
W-184  15  DEN=5.9500 00.3655706736932543 873 END
W-186  15  DEN=5.5484 00.3655706736932543 873 END
B-10   15  DEN=2.37   00.6344293263067458 873 END
W-182  16  DEN=5.1022 00.1836496437525630 873 END
W-183  16  DEN=2.7742 00.1836496437525630 873 END
W-184  16  DEN=5.9500 00.1836496437525630 873 END
W-186  16  DEN=5.5484 00.1836496437525630 873 END
B-10   16  DEN=2.37   00.8163503562474370 873 END
Pb     17  DEN=11.344 00.9997035175329104 873 END
B-10   17  DEN=2.37   00.0002964824670896 873 END
ans =
    1.0000    8.7731
    2.0000   23.9255
   15.0000   54.4823
   16.0000   82.7336
   17.0000   84.3561
   10.0000  135.6530
```

APPENDIX E
E MATLAB OBCTURBOMACHINE CALCS

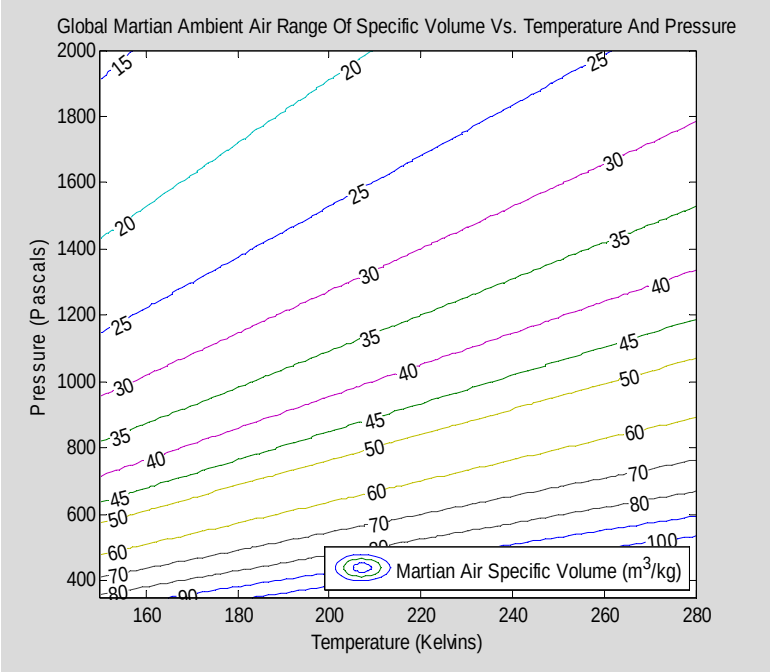
This appendix contains the MATLAB code in the MSWORD/MATLAB Notebook format which provides the capability for self-documenting calculations. M-file subroutines for fluid properties as defined in Appendix C are used freely herein. Some of the plots generated in this appendix through the calculations here have been cut/pasted into other locations, where they have been assigned captions.

```
R=191.06;
p=linspace(350,2000);
t=linspace(150,280);
[P,T]=meshgrid(p,t);
Rho=P./T/R*1000;
[C,h]=contour(T,P,Rho,5:5:75);
clabel(C,h);
title('Global Martian Ambient Air Range Of Densities Vs. Temperatures
And Pressures')
xlabel('Temperature (Kelvins)')
ylabel('Pressure (Pascals)')
legend('Martian Air Density (grams/meter^3)','Location','SouthEast')
```

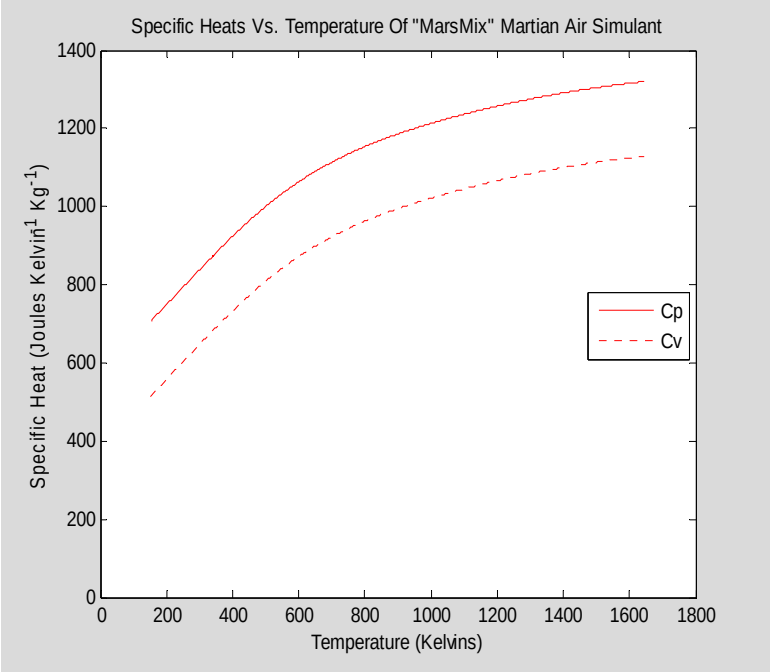



```
Vsp=1000./Rho;
[C,h]=contour(T,P,Vsp,[15:5:50 60:10:100]);
clabel(C,h);
```

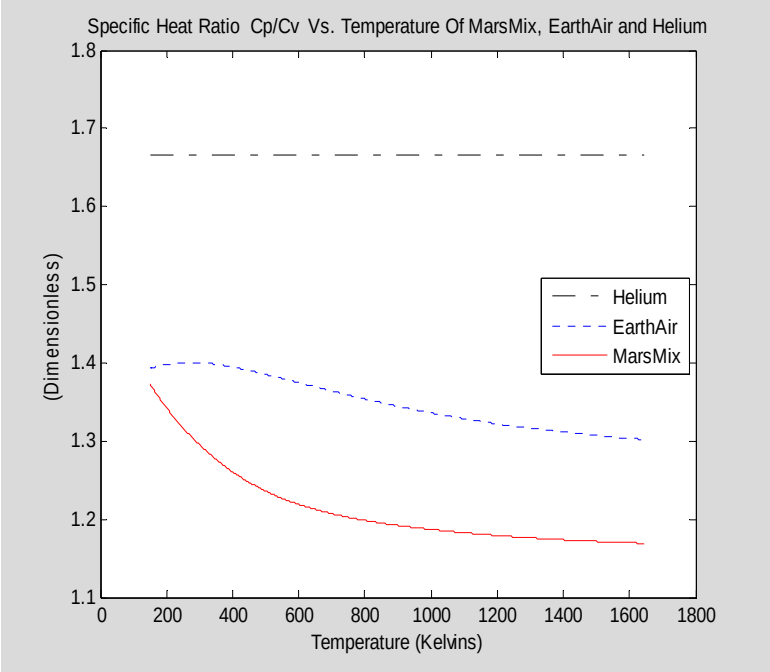
```
title('Global Martian Ambient Air Range Of Specific Volume Vs.
Temperature And Pressure')
xlabel('Temperature (Kelvins)')
ylabel('Pressure (Pascals)')
legend('Martian Air Specific Volume (m^3/kg)','Location','SouthEast')
```



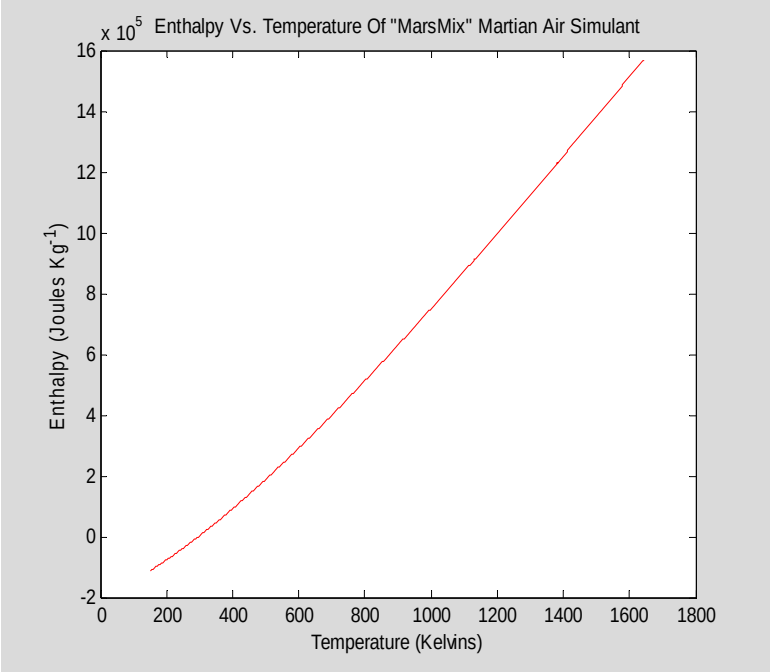

```
clear
R=191.06;
T=150:1643;
Cp=SpecificHeat('MarsMix',T);
Cv=Cp-R;
plot(T,Cp,'-r',T,Cv,':r')
title('Specific Heats Vs. Temperature Of "MarsMix" Martian Air
Simulant')
xlabel('Temperature (Kelvins)')
ylim([0 1400])
ylabel('Specific Heat (Joules Kelvin^-^1 Kg^-^1)')
legend('Cp','Cv','Location','East')
```



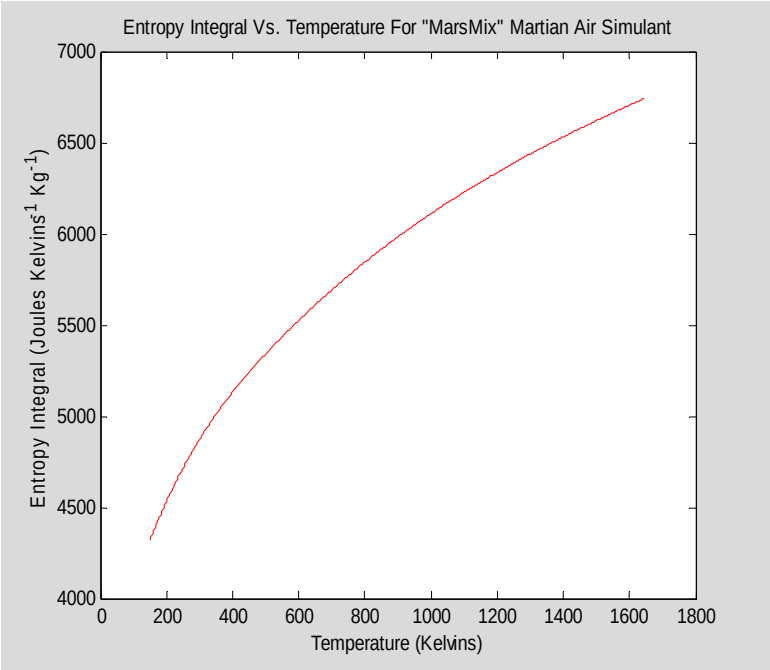

```
kappa_MarsMix=Cp./Cv;
Cp_EarthAir=SpecificHeat('EarthAir',T);
Cv_EarthAir=Cp_EarthAir-GasConstant('EarthAir');
kappa_EarthAir=Cp_EarthAir./Cv_EarthAir;
Cp_He=SpecificHeat('He',T);
Cv_He=Cp_He-GasConstant('He');
kappa_He=Cp_He./Cv_He;
plot(T,kappa_He,'-.k',T,kappa_EarthAir,':b',T,kappa_MarsMix,'-r')
title('Specific Heat Ratio  Cp/Cv  Vs. Temperature Of MarsMix, EarthAir
and Helium')
xlabel('Temperature (Kelvins)')
ylabel('(Dimensionless)')
legend('Helium','EarthAir','MarsMix','Location','East')
```



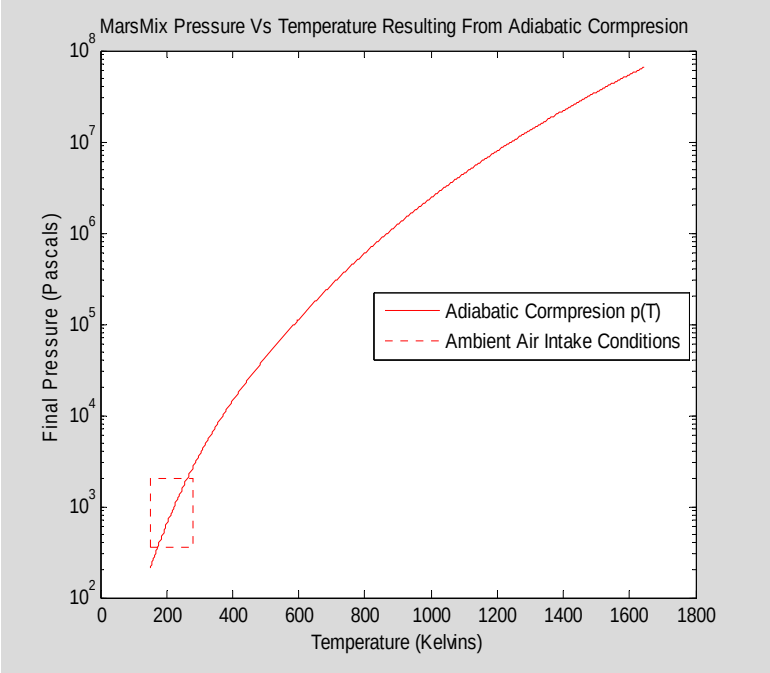

```
h=Enthalpy('MarsMix',T);
plot(T,h,'r')
xlim([0 1800])
ylabel('Enthalpy (Joules Kg^-^1)')
xlabel('Temperature (Kelvins)')
title('Enthalpy Vs. Temperature Of "MarsMix" Martian Air Simulant')
```




```
S0=EntropyIntegral('MarsMix',T);
plot(T,S0,'r')
title('Entropy Integral Vs. Temperature For "MarsMix" Martian Air
Simulant')
xlim([0 1800])
ylabel('Entropy Integral (Joules Kelvins^-^1 Kg^-^1)')
xlabel('Temperature (Kelvins)')
```

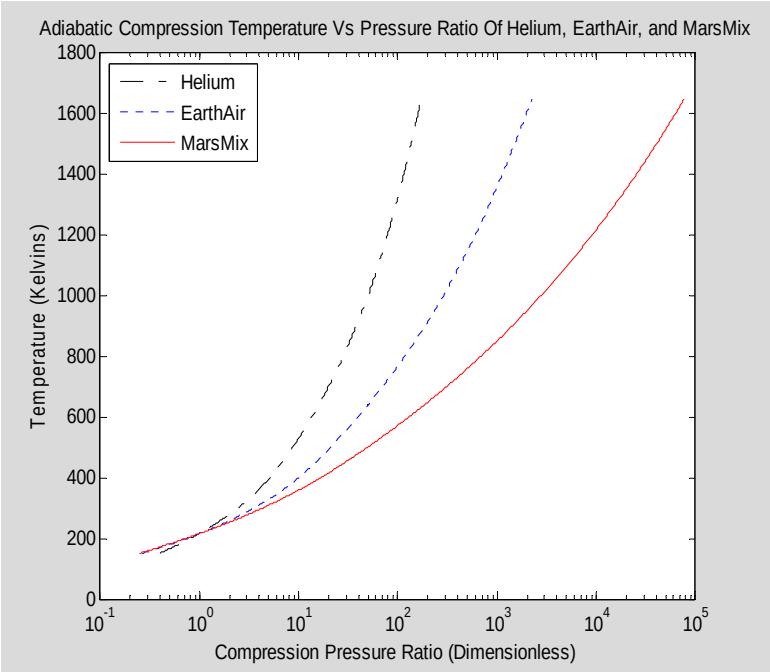


```
clf;
iref=find(T>214.5, 1);
p_adiabatic=850*exp((S0-S0(iref))/R);
T_ambient=[150 280 280 150 150];
p_ambient=[350 350 2000 2000 350];
semilogy(T,p_adiabatic,'r',T_ambient,p_ambient,':r')
xlim([0 1800])
legend('Adiabatic Cormpresion p(T)','Ambient Air Intake Conditions
','Location','East')
ylabel('Final Pressure (Pascals)')
xlabel('Temperature (Kelvins)')
title('MarsMix Pressure Vs Temperature Resulting From Adiabatic
Cormpresion ')
```

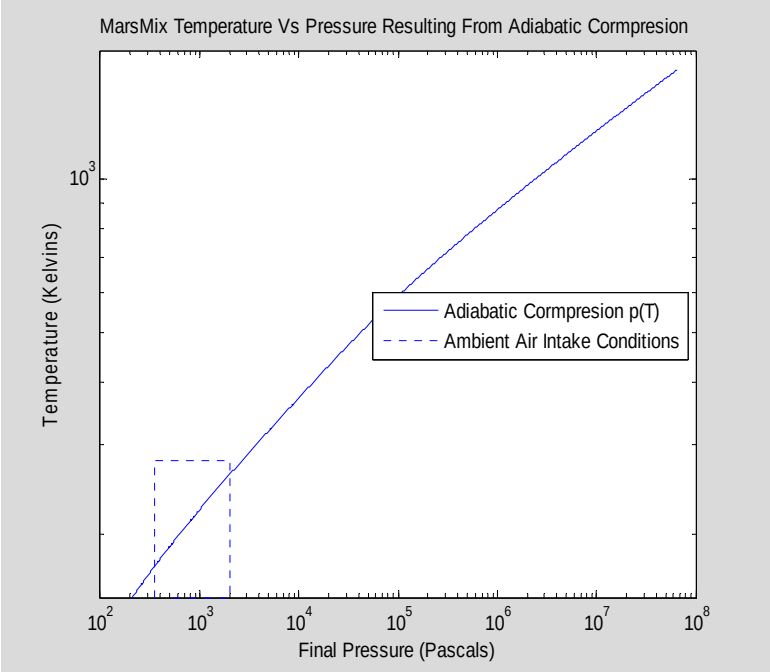



```
Gas='MarsMix';
Pratio_MarsMix=exp((EntropyIntegral(Gas,T)-
EntropyIntegral(Gas,215))/GasConstant(Gas));
Gas='EarthAir';
Pratio_EarthAir=exp((EntropyIntegral(Gas,T)-
EntropyIntegral(Gas,215))/GasConstant(Gas));
Gas='He';
Pratio_Helium=exp((EntropyIntegral(Gas,T)-
EntropyIntegral(Gas,215))/GasConstant(Gas));
semilogx(Pratio_Helium,T,'-.k',
Pratio_EarthAir,T,':b',Pratio_MarsMix,T,'r')
ylim([0 1800])
legend('Helium','EarthAir','MarsMix','Location','NorthWest')
```

```
ylabel('Temperature (Kelvins)')
xlabel('Compression Pressure Ratio (Dimensionless)');
title('Adiabatic Compression Temperature Vs Pressure Ratio Of Helium,
EarthAir, and MarsMix')
```



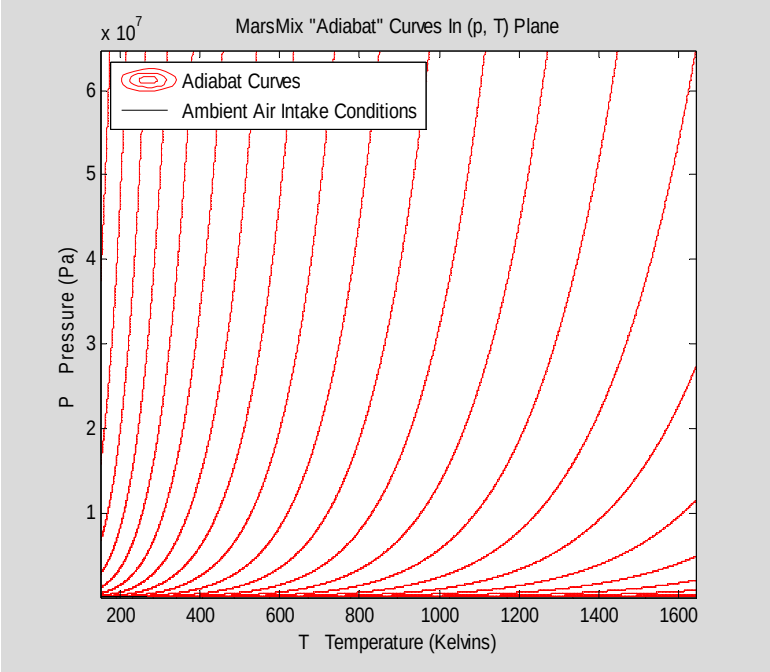

```
loglog(p_adiabatic,T,'b',p_ambient,T_ambient,':b')
ylim([0 1800])
legend('Adiabatic Cormpresion p(T)','Ambient Air Intake Conditions
','Location','East')
xlabel('Final Pressure (Pascals)')
ylabel('Temperature (Kelvins)')
title('MarsMix Temperature Vs Pressure Resulting From Adiabatic
Cormpresion ')
```



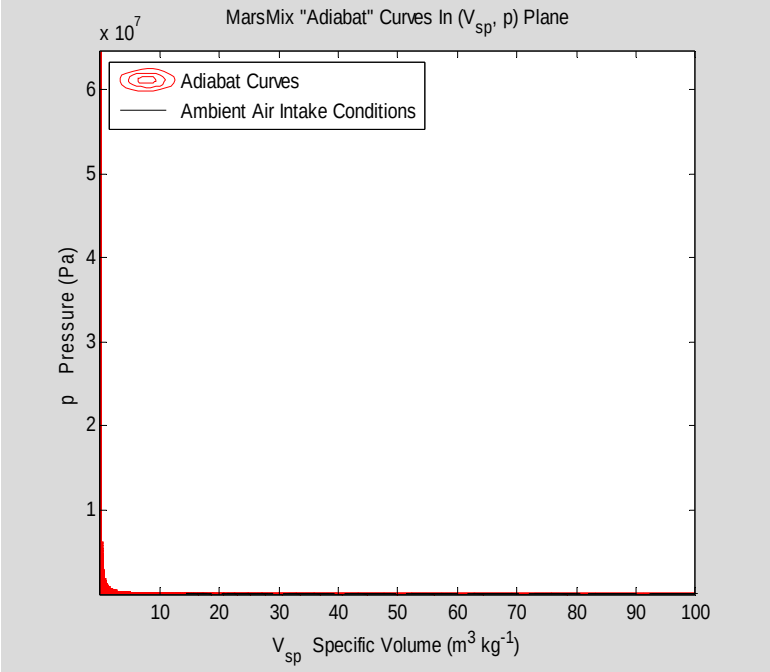
$$p(T) = 850 \exp\left(\frac{S_0(T) - S_0(215)}{R}\right)$$

$$\frac{R\ln(p(T)) - S_0(T)}{R\ln(850) - S_0(215)} = 1$$

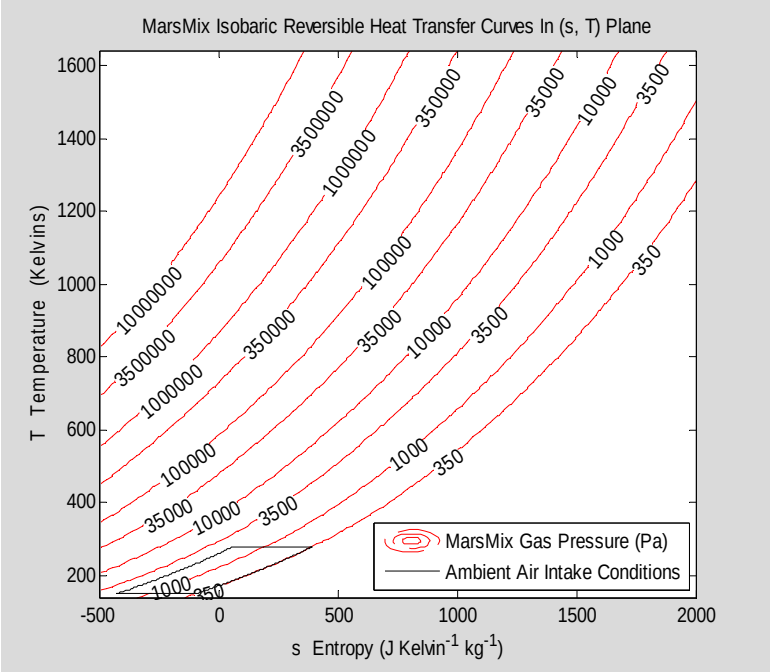
```
[P_mesh,T_mesh]=meshgrid(p_adiabatic,T);
F=(R*log(P_mesh)-EntropyIntegral('MarsMix',T_mesh))/(R*log(850)-
S0(iref));
contour(T_mesh,P_mesh,F,0.20:0.05:1.80,'r')
ylabel('P   Pressure (Pa)')
xlabel('T   Temperature (Kelvins)')
title('MarsMix "Adiabat" Curves In (p, T) Plane')
hold on
plot(T_ambient,p_ambient,'--k')
hold off
legend('Adiabat Curves','Ambient Air Intake
Conditions','Location','NorthWest')
```



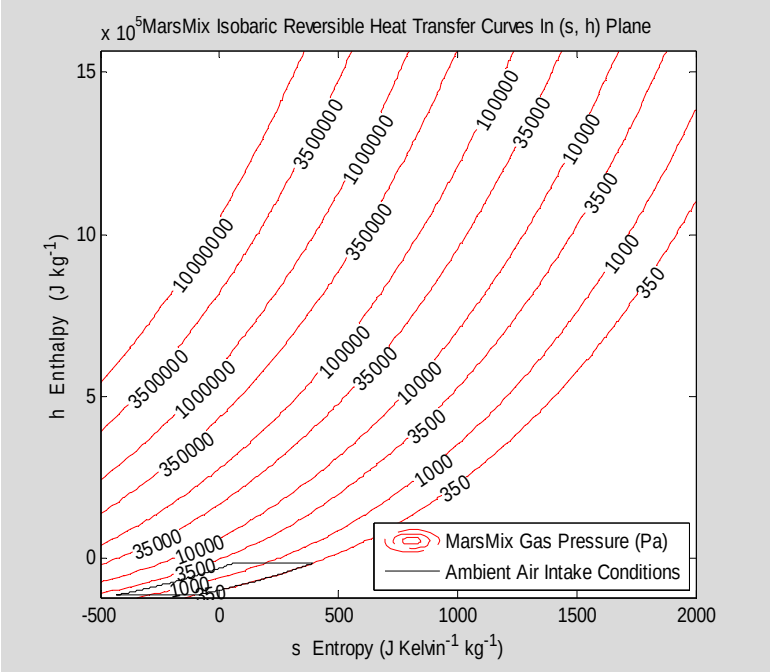

```
Vsp=10.^linspace(-2, 2);
[P_mesh,Vsp_mesh]=meshgrid(p_adiabatic,Vsp);
T_mesh=P_mesh.*Vsp_mesh/R;
F=(R*log(P_mesh)-EntropyIntegral('MarsMix',T_mesh))/(R*log(850)-
S0(iref));
contour(Vsp_mesh,P_mesh,F,0.20:0.05:1.80,'r')
ylabel('p   Pressure (Pa)')
xlabel('V_s_p  Specific Volume (m^3 kg^-^1)')
title('MarsMix "Adiabat" Curves In (V_s_p, p) Plane');
hold on
p_start=linspace(p_ambient(1),p_ambient(2));
T_start=linspace(T_ambient(1),T_ambient(2));
for i=2:4
p_start=[p_start  linspace(p_ambient(i),p_ambient(i+1))];
T_start=[T_start  linspace(T_ambient(i),T_ambient(i+1))];
end
Vsp_start=R*T_start./p_start;
plot(Vsp_start,p_start,'--k')
hold off
legend('Adiabat Curves','Ambient Air Intake Conditions',
'Location','NorthWest')
```




```
s_scale=linspace(-500,2000);
T_scale=linspace(140,1643);
[s_sTmesh,T_sTmesh]=meshgrid(s_scale,T_scale);
FsT_heataddition=850*exp((EntropyIntegral('MarsMix',T_sTmesh)-
EntropyIntegral('MarsMix',215)-s_sTmesh)/R);
contour_pressures=[350 1000 3500 10000 35000 100000 350000 1e6 3.5e6
1e7];
[C,h]=contour(s_sTmesh,T_sTmesh,FsT_heataddition,contour_pressures,'--
r');
clabel(C,h);
xlabel('s  Entropy (J Kelvin^-^1 kg^-^1)')
ylabel('T  Temperature  (Kelvins)')
title('MarsMix Isobaric Reversible Heat Transfer Curves In (s, T)
Plane')
s_start=EntropyIntegral('MarsMix',T_start)-
EntropyIntegral('MarsMix',215)-R*log(p_start/850);
hold on
plot(s_start,T_start,'-k')
hold off
legend('MarsMix Gas Pressure (Pa)','Ambient Air Intake
Conditions','Location','SouthEast')
```




```
T_interp=130:1650;
h_interp=Enthalpy('MarsMix',T_interp);
s_scale=linspace(-500,2000);
h_scale=linspace(Enthalpy('MarsMix',140),Enthalpy('MarsMix',1643));
[s_shmesh,h_shmesh]=meshgrid(s_scale,h_scale);
T_shmesh=interp1(h_interp,T_Interp,h_shmesh);
Fsh_heataddition=850*exp((EntropyIntegral('MarsMix',T_shmesh)-
EntropyIntegral('MarsMix',215)-s_sTmesh)/R);
contour_pressures=[350 1000 3500 10000 35000 100000 350000 1e6 3.5e6
1e7];
[C,h]=contour(s_sTmesh,h_shmesh,Fsh_heataddition,contour_pressures,'--
r');
clabel(C,h);
xlabel('s  Entropy (J Kelvin^-^1 kg^-^1)')
ylabel('h  Enthalpy  (J kg^-^1)')
title('MarsMix Isobaric Reversible Heat Transfer Curves In (s, h)
Plane')
s_start=EntropyIntegral('MarsMix',T_start)-
EntropyIntegral('MarsMix',215)-R*log(p_start/850);
h_start=Enthalpy('MarsMix',T_start);
hold on
plot(s_start,h_start,'-k')
hold off
legend('MarsMix Gas Pressure (Pa)','Ambient Air Intake
Conditions','Location','SouthEast')
```



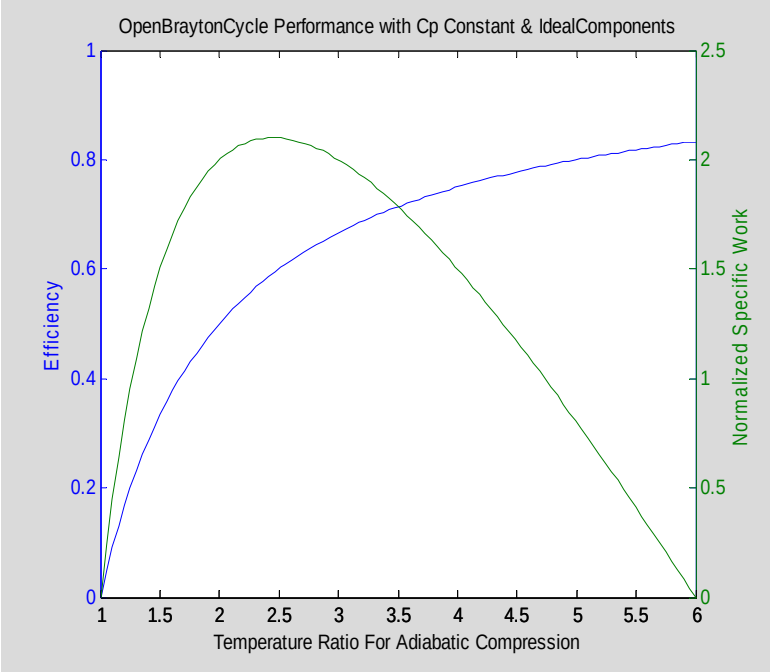
For analyses of the ideal Open Brayton Cycle with constant specific heats, the efficiency and the specific work are as follows:

$$\eta \equiv 1 - \frac{1}{\tau_c}$$

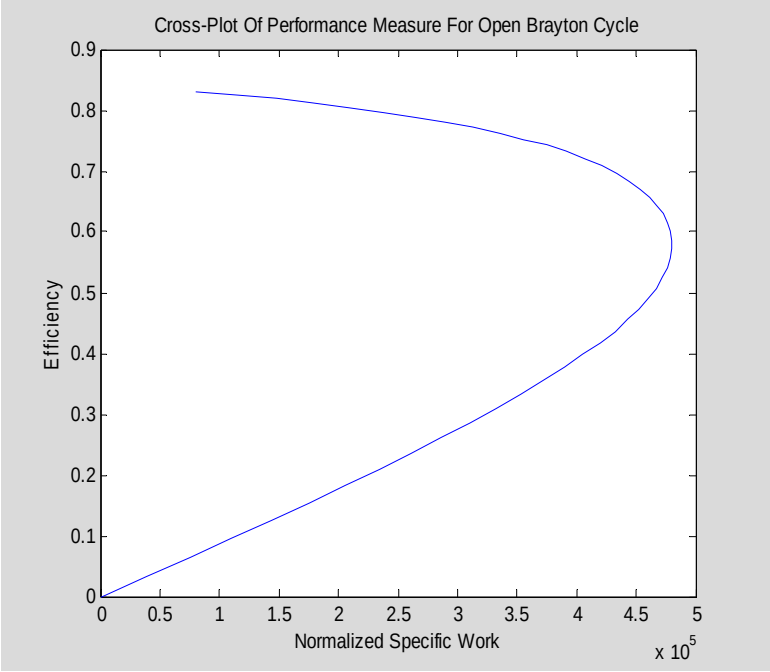
$$w = C_p T_0 \left(\frac{\tau}{\tau_c} \right) (\tau_c - 1)$$

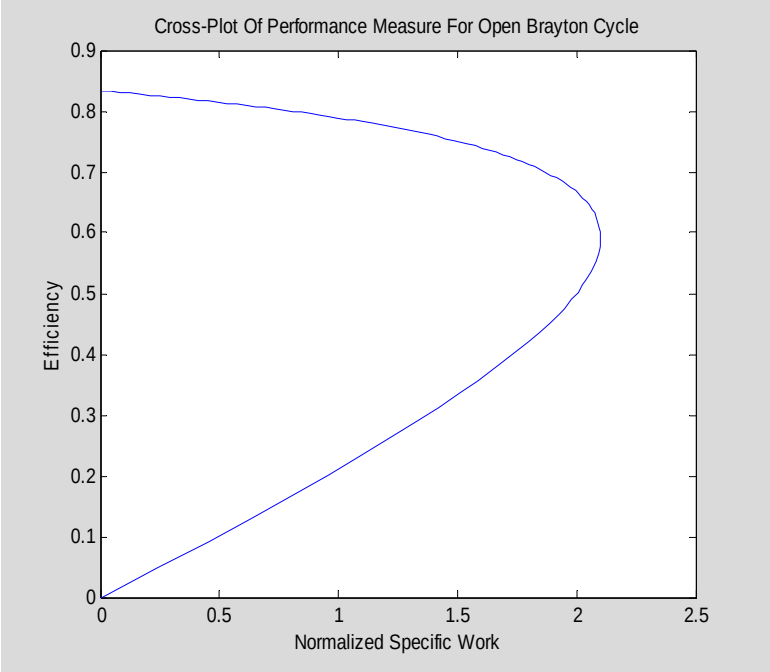
where τ_c is the compression temperature ratio and where τ is the overall temperature ratio. Plotting this we have as follows:

```
tau=6;tauc=linspace(1,tau);
efficiency=1-1./tauc;
wd=((tau./tauc)-1).*(tauc-1);
[AX,H1,H2]=plotyy(tauc,efficiency,tauc,wd);
set(get(AX(1),'Ylabel'),'String','Efficiency')
set(get(AX(2),'Ylabel'),'String','Normalized Specific Work')
xlabel('Temperature Ratio For Adiabatic Compression')
title('OpenBraytonCycle Performance with Cp Constant &
IdealComponents')
```



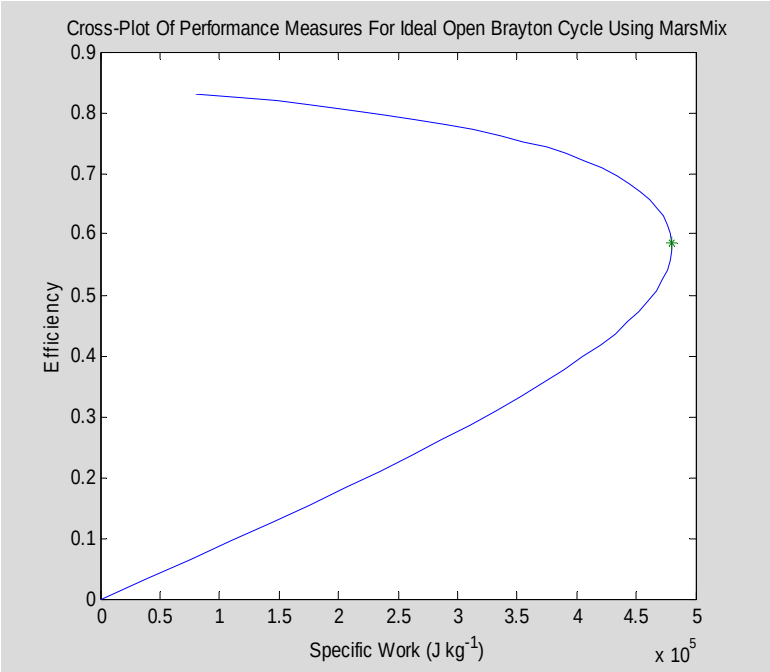

```
plot(wd,efficiency)
xlabel('Normalized Specific Work'),ylabel('Efficiency')
title('Cross-Plot Of Performance Measure For Open Brayton Cycle')
```



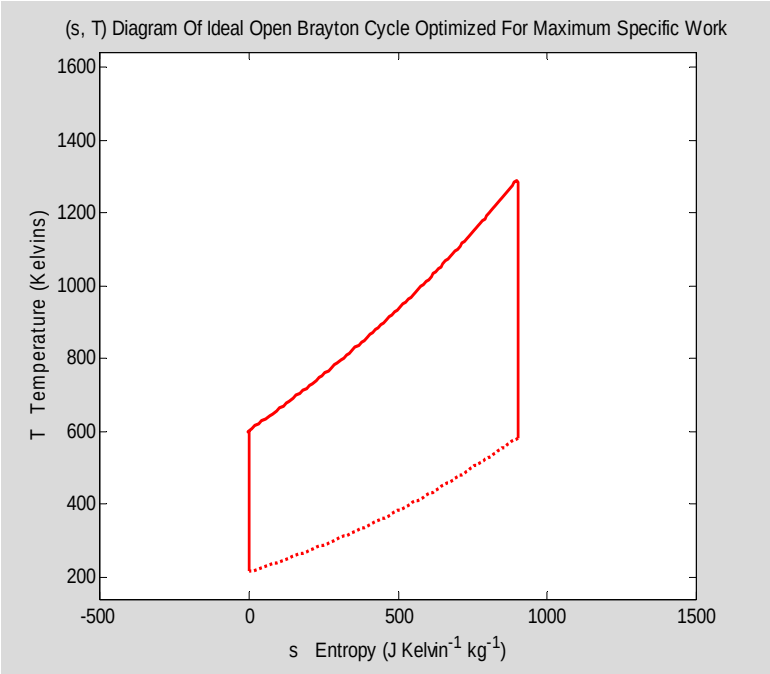



```
clear
R=GasConstant('MarsMix');
T=150:1643;
S0=EntropyIntegral('MarsMix',T);
pic=logspace(0,4);
T0=215;T4=1290;
T2=interp1(S0,T,EntropyIntegral('MarsMix',T0)+R*log(pic));
T5=interp1(S0,T,EntropyIntegral('MarsMix',T4)-R*log(pic));
qdotpermdot=Enthalpy('MarsMix',T4)-Enthalpy('MarsMix',T2);
Ppermdot=Enthalpy('MarsMix',T0)+Enthalpy('MarsMix',T4)-
Enthalpy('MarsMix',T2)-Enthalpy('MarsMix',T5);
efficiency=Ppermdot./qdotpermdot;
wd=Ppermdot;
tauc=T2/T0;
i_opt=find(wd==max(wd))
T2opt=T2(i_opt)
T5opt=T5(i_opt)
plot(wd,efficiency,wd(27),efficiency(i_opt),'*')
xlim([0 5e5]);
xlabel('Specific Work (J kg^-^1)'),ylabel('Efficiency')
title('Cross-Plot Of Performance Measures For Ideal Open Brayton Cycle
Using MarsMix')

i_opt =
    27
T2opt =
  600.9912
```

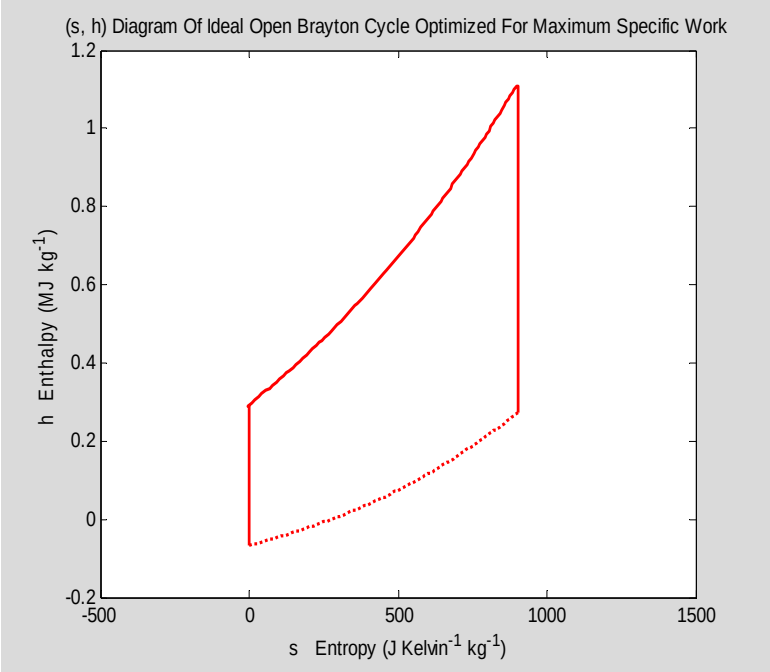
```
T5opt =
  583.0923
```



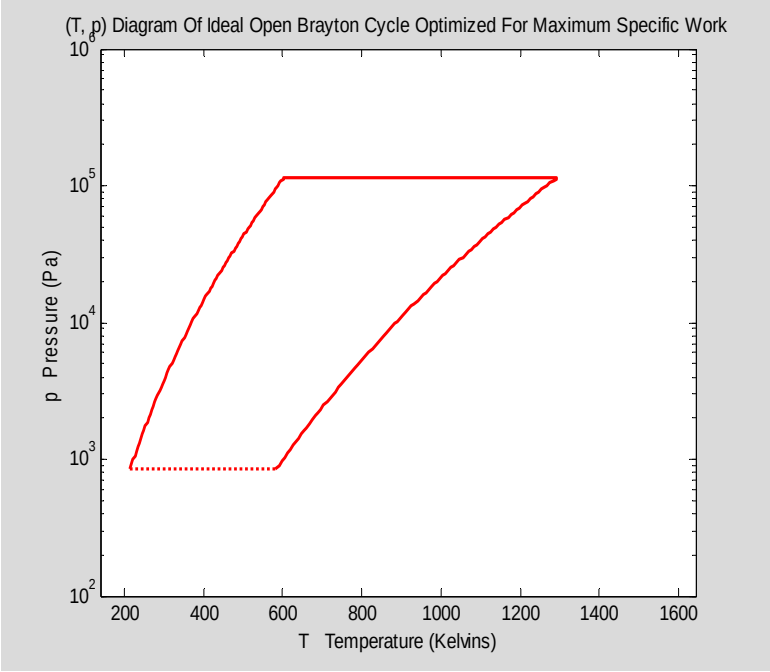

```
picopt=pic(i_opt);
p0=850;
s0=0;s4=EntropyIntegral('MarsMix',T4)-EntropyIntegral('MarsMix',T2opt);
sobc=[zeros(1,100) EntropyIntegral('MarsMix',linspace(T2opt,T4))-
EntropyIntegral('MarsMix',T2opt)-log(picopt) s4*ones(1,100)];
sext=EntropyIntegral('MarsMix',linspace(T5opt,T0))-
EntropyIntegral('MarsMix',T0);

Tobc=[linspace(T0,T2opt)  linspace(T2opt,T4)  linspace(T4,T5opt)];
Text=linspace(T5opt,T0);
hobc=Enthalpy('MarsMix',Tobc);
hext=Enthalpy('MarsMix',Text);
pobc=p0*exp((EntropyIntegral('MarsMix',Tobc)-
EntropyIntegral('MarsMix',T0)-(sobc-s0))/R);
pext=p0*exp((EntropyIntegral('MarsMix',Text)-
EntropyIntegral('MarsMix',T0)-(sext-s0))/R);
plot(sobc,Tobc,'r',sext,Text,':r','Linewidth',2)
xlim([-500 1500])
ylim([140 1643])
xlabel('s   Entropy (J Kelvin^-^1 kg^-^1)')
ylabel('T  Temperature (Kelvins)')
title('(s, T) Diagram Of Ideal Open Brayton Cycle Optimized For Maximum
Specific Work')
```




```
plot(sobc,hobc/1e6,'r',sext,hext/1e6,':r','Linewidth',2)
xlim([-500 1500])
xlabel('s   Entropy (J Kelvin^-^1 kg^-^1)')
ylabel('h  Enthalpy (MJ kg^-^1)')
title('(s, h) Diagram Of Ideal Open Brayton Cycle Optimized For Maximum
Specific Work')
```




```
semilogy(Tobc,pobc,'r',Text,pext,':r','LineWidth',2)
xlim([140 1643])
xlabel('T   Temperature (Kelvins)')
ylabel('p  Pressure (Pa)')
title('(T, p) Diagram Of Ideal Open Brayton Cycle Optimized For Maximum
Specific Work')
```



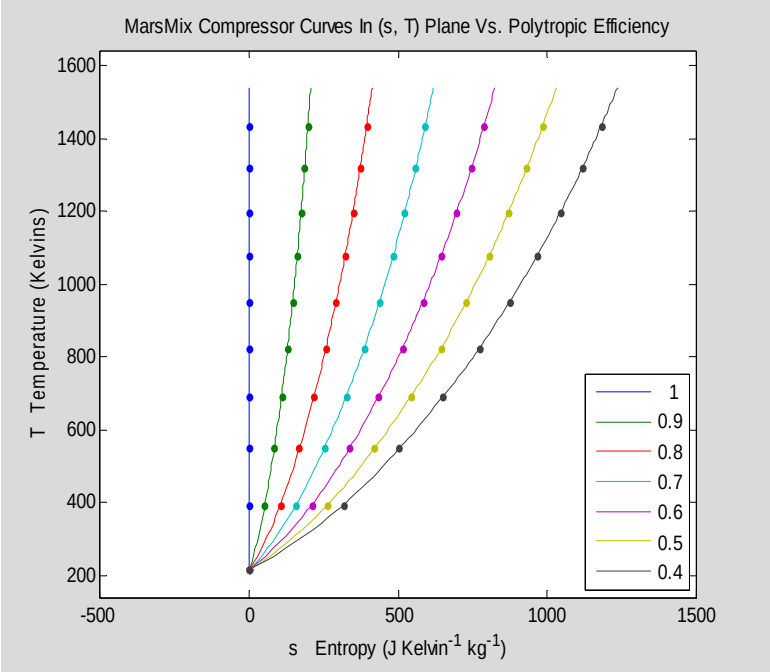
For a lossy compressor working with a variable specific heat gas, the equations defining a polytrope curve are as follows:

$$\left\{\begin{array}{l} s = s_0 + (1-e_c)(S_0[T] - S_0[T_0]) \\ h = h[T] \\ p = p_0 \exp\left(e_c \dfrac{S_0[T] - S_0[T_0]}{R}\right) \end{array}\right\}_{\text{compressor}}$$

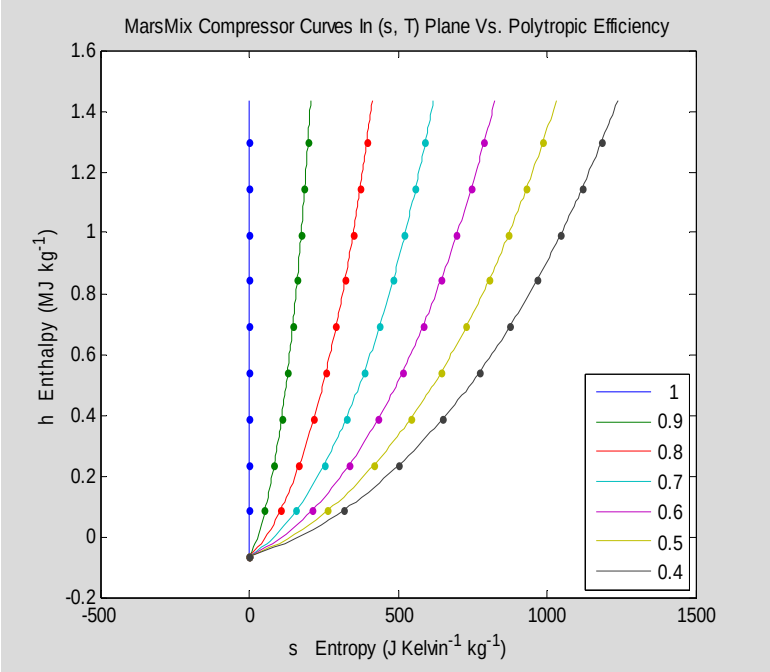
A set of polytrope curves are generated and plotted in the following MATLAB cells. The starting point is to specify the total work added (w) and to set it equal to the change in enthalpy from an initial reference state. Th eomputational algorithm is then as follows:

$$h_* = h[T_0] + w$$
$$T_* = h^{-1}[h_*]$$
$$s_* = s_0 + (1-e_c)(S_0[T_*] - S_0[T_0])$$
$$p = p_0 \exp\left(e_c \frac{S_0[T_*] - S_0[T_0]}{R}\right)$$

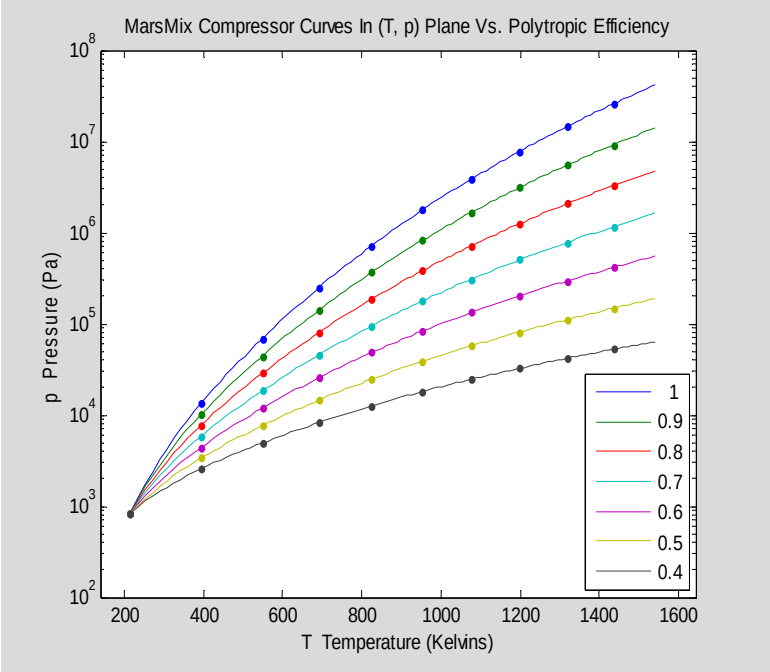
```
T=140:1643;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);R=Ga
sConstant('MarsMix');
T0=215;T4=1290;p0=850;s0=0;h0=Enthalpy('MarsMix',T0);
w=linspace(0,1.5e6)';
ec=[1 0.90 0.80 0.70 0.60 0.50 0.40];
hpoly=h0+w;
Tpoly=interp1(h,T,hpoly);
deltaS0=EntropyIntegral('MarsMix',Tpoly)-EntropyIntegral('MarsMix',T0);
spoly=s0+deltaS0*(1-ec);
p=p0*exp(deltaS0*ec/R);
plot(spoly,Tpoly,spoly(1:10:end,:),Tpoly(1:10:end ),'.')
xlim([-500 1500])
ylim([140 1643])
title('MarsMix Compressor Curves In (s, T) Plane Vs. Polytropic
Efficiency')
xlabel('s   Entropy (J Kelvin^-^1 kg^-^1)')
ylabel('T  Temperature (Kelvins)')
legend(num2str(ec'),'Location','SouthEast')
```




```
plot(spoly,hpoly/1e6,spoly(1:10:end,:),hpoly(1:10:end )/1e6,'.')
xlim([-500 1500])
ylim([-.2 1.6])
title('MarsMix Compressor Curves In (s, T) Plane Vs. Polytropic
Efficiency')
xlabel('s   Entropy (J Kelvin^-^1 kg^-^1)')
ylabel('h  Enthalpy (MJ kg^-^1)')
legend(num2str(ec'),'Location','SouthEast')
```




```
semilogy(Tpoly,p,Tpoly(1:10:end ),p(1:10:end,:),'.')
xlim([140 1643])
title('MarsMix Compressor Curves In (T, p) Plane Vs. Polytropic
Efficiency')
ylabel('p  Pressure (Pa)')
xlabel('T  Temperature (Kelvins)')
legend(num2str(ec'),'Location','SouthEast')
```



For expanders such as turbines the polytropic equations to use are as follows:

$$\left\{\begin{array}{l} s = s_0 + \left(\dfrac{1-e_E}{e_E}\right)\left(S_0[T] - S_0[T]\right) \\ h = h[T] \\ p = p_0 \exp\left(\dfrac{1}{e_E}\dfrac{S_0[T]-S_0[T_0]}{R}\right) \end{array}\right\}_{\text{expander}}$$

The mathematical computation will be as follows:

$$s_4 = s_0 + S_0[T_4] - S_0[T_0] - R\ln\left(\frac{p_4}{p_0}\right)$$

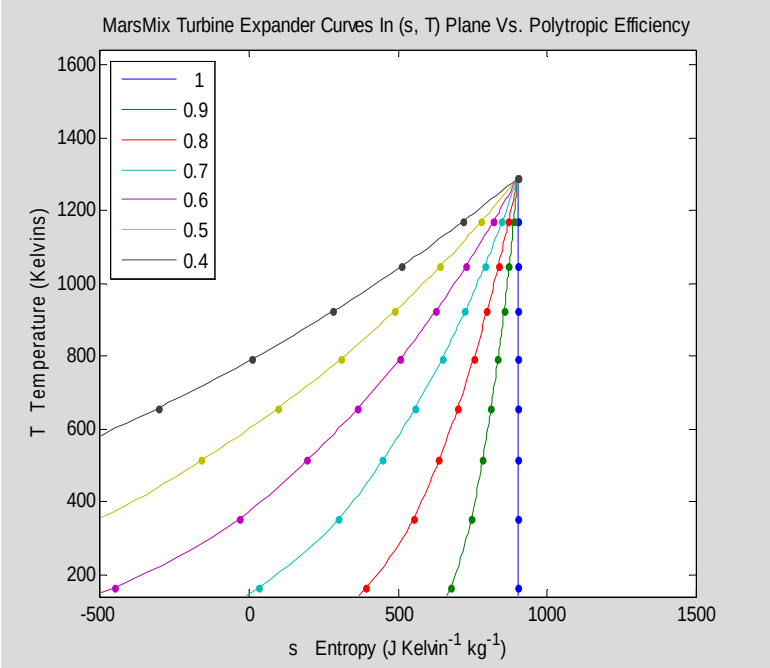
$$h_* = h[T_4] - w$$

$$T_* = h^{-1}[h_*]$$

$$s_* = s_4 - \left(\frac{1-e_E}{e_E}\right)\left(S_0[T_4] - S_0[T_*]\right)$$

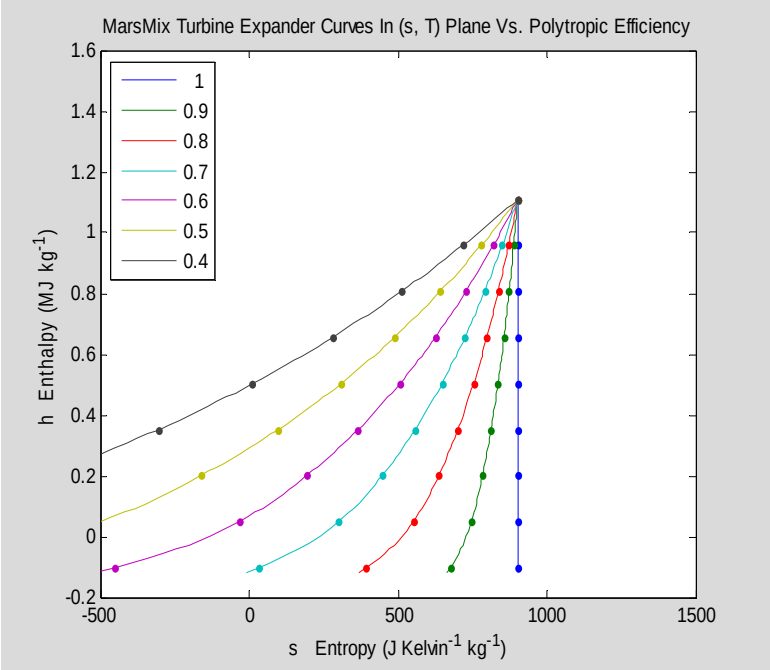
$$p = p_4 \exp\left(\left(\frac{1}{e_E}\right)\frac{S_0[T_4]-S_0[T_*]}{R}\right)$$

```
clear;
T=140:1643;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);R=Ga
sConstant('MarsMix');
T0=215;T4=1290;p0=850;p4=112690;s0=0;h4=Enthalpy('MarsMix',T4);
s4=s0+EntropyIntegral('MarsMix',T4)-EntropyIntegral('MarsMix',T0)-
R*log(p4/p0);
w=linspace(0,1.5e6)';
eE=[1 0.90 0.80 0.70 0.60 0.50 0.40];
hpoly=h4-w;
Tpoly=interp1(h,T,hpoly);
deltaS0=EntropyIntegral('MarsMix',T4)-EntropyIntegral('MarsMix',Tpoly);
spoly=s4-(deltaS0*((1-eE)./eE));
p=p4*exp(-deltaS0*(ones(size(eE))./eE)/R);
plot(spoly,Tpoly,spoly(1:10:end,:),Tpoly(1:10:end ),'.')
xlim([-500 1500])
ylim([140 1643])
title('MarsMix Turbine Expander Curves In (s, T) Plane Vs. Polytropic
Efficiency')
xlabel('s   Entropy (J Kelvin^-^1 kg^-^1)')
ylabel('T  Temperature (Kelvins)')
legend(num2str(eE'),'Location','NorthWest')
```

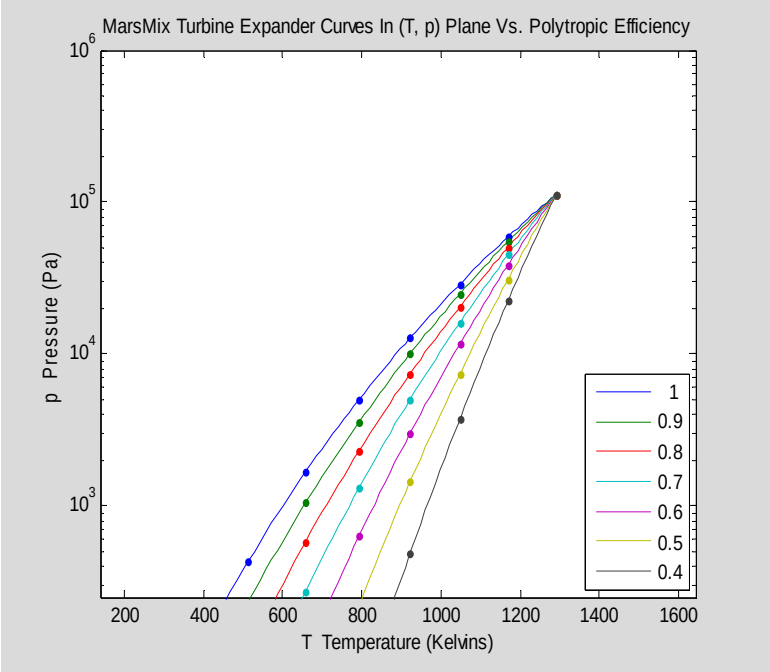



```
plot(spoly,hpoly/1e6,spoly(1:10:end,:),hpoly(1:10:end )/1e6,'.')
xlim([-500 1500])
ylim([-.2 1.6])
title('MarsMix Turbine Expander Curves In (s, T) Plane Vs. Polytropic
Efficiency')
xlabel('s   Entropy (J Kelvin^-^1 kg^-^1)')
ylabel('h  Enthalpy (MJ kg^-^1)')
```

```
legend(num2str(eE'),'Location','NorthWest')
```




```
semilogy(Tpoly,p,Tpoly(1:10:end ),p(1:10:end,:),'.')
xlim([140 1643])
ylim([250 1e6])
title('MarsMix Turbine Expander Curves In (T, p) Plane Vs. Polytropic
Efficiency')
ylabel('p  Pressure (Pa)')
xlabel('T  Temperature (Kelvins)')
legend(num2str(eE'),'Location','SouthEast')
```



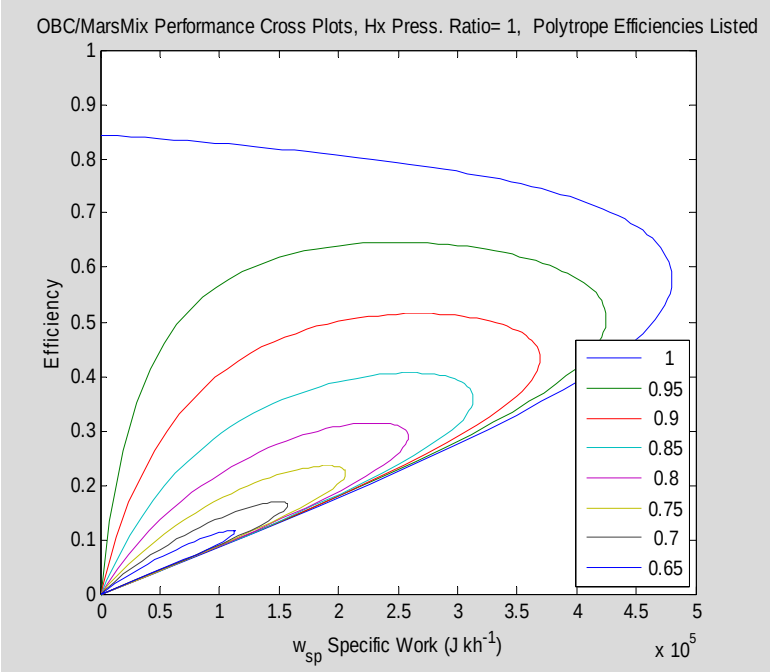
To investigate the performance of the Open Brayton Cycle in MarsMix gas with lossy components, we implement the following formulae:

$$h_0 = h[T_0]$$
$$h_4 = h[T_4]$$
$$h_2 = h_0 + (-w_c)$$
$$T_2 = h^{-1}[h_2]$$
$$p_2 = p_0 \exp\left(e_c \frac{S_0[T_2] - S_0[T_0]}{R} \right)$$
$$p_4 = e_p p_2$$
$$T_5 = S_0^{-1}\left[S_0[T_4] - e_E R \ln\left(\frac{p_4}{p_0} \right) \right]$$
$$h_5 = h[T_5]$$
$$w_E = h_4 - h_5$$
$$w_{sp} = w_E - (-w_c)$$
$$(\dot{q}/\dot{m}) = h_4 - h_2$$
$$\eta = \frac{w_{sp}}{(\dot{q}/\dot{m})}$$

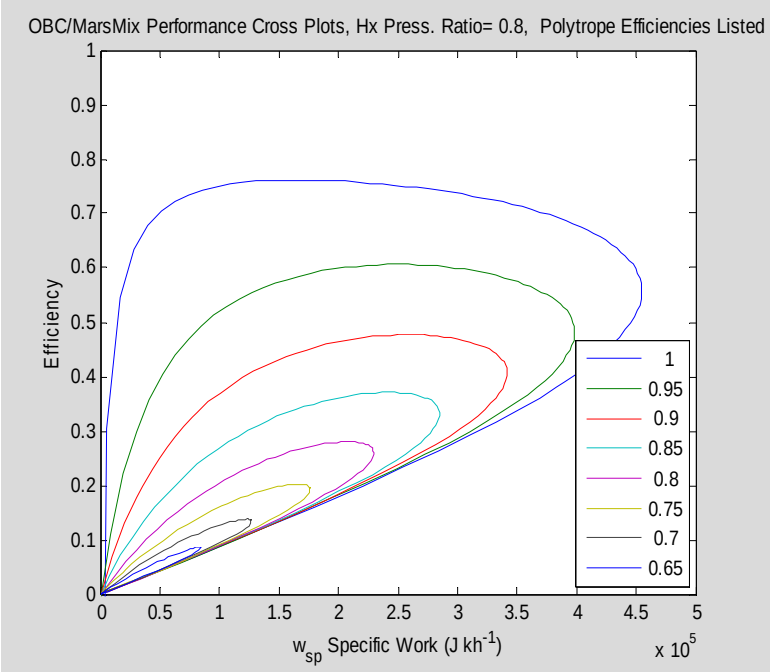
For these calculatios the limiting temperatures will be held fixed at 215 and 1290 Kelvins. The compressor specific work variable, ($-w_c$), will be given an entire vector of successive values in order to generate a performance curve. However, there are also three outer input values, i.e., the efficiencies e_c, e_E, and e_p.

```
clear;
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);R=Ga
sConstant('MarsMix');
T0=215;T4=1290;p0=850;
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);
wcm=linspace(0,1.4e6)';
h2=h0+wcm;
T2=interp1(h,T,h2);

ec=[1 0.95 0.9 0.85 0.8 0.75 0.7 0.65];eE=ec;ep=1.0;
p2=p0*exp((interp1(T,S0,T2)-S00)*ec/R);
p4=p2*ep;
T5=interp1(S0,T,S04-log(p4/p0).* (ones(size(wcm)) *eE)*R);
h5=interp1(T,h,T5);
we=h4-h5;
wsp=we-wcm*ones(size(ec));
```

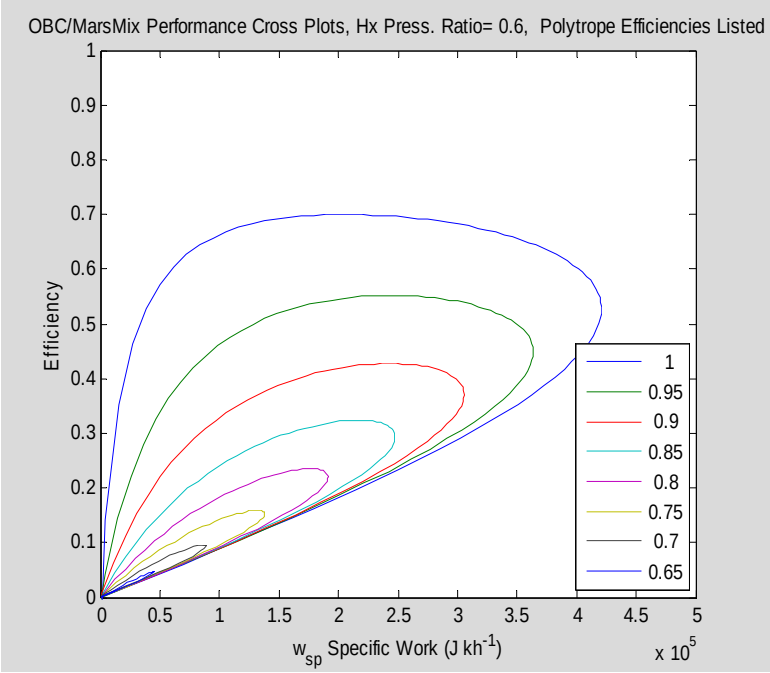
```
efficiency=wsp./((h4-h2)*ones(size(ec)));
plot(wsp,efficiency)
xlim([0 5e5])
ylim([0 1])
xlabel('w_s_p Specific Work (J kh^-^1)')
ylabel('Efficiency')
title(['OBC/MarsMix Performance Cross Plots, Hx Press. Ratio= '
num2str(ep) ',  Polytrope Efficiencies Listed']),
legend(num2str(ec'),'Location','SouthEast')
```



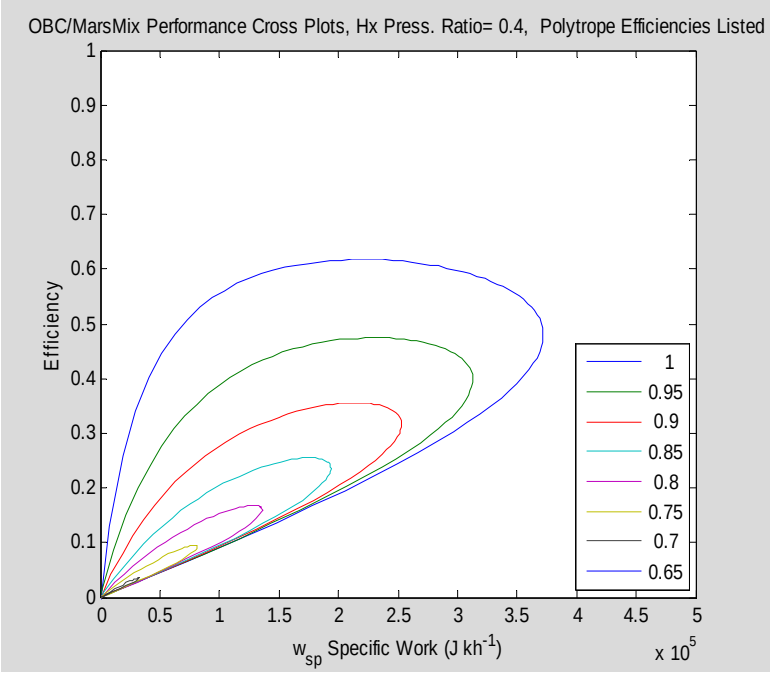

```
ec=[1 0.95 0.9 0.85 0.8 0.75 0.7 0.65];eE=ec;ep=0.8;
p2=p0*exp((interp1(T,S0,T2)-S00)*ec/R);
p4=p2*ep;
T5=interp1(S0,T,S04-log(p4/p0).* (ones(size(wcm)) *eE)*R);
h5=interp1(T,h,T5);
we=h4-h5;
wsp=we-wcm*ones(size(ec));
efficiency=wsp./((h4-h2)*ones(size(ec)));
plot(wsp,efficiency)
xlim([0 5e5])
ylim([0 1])
xlabel('w_s_p Specific Work (J kh^-^1)')
ylabel('Efficiency')
title(['OBC/MarsMix Performance Cross Plots, Hx Press. Ratio= '
num2str(ep) ',  Polytrope Efficiencies Listed']),
legend(num2str(ec'),'Location','SouthEast')
```




```
ec=[1 0.95 0.9 0.85 0.8 0.75 0.7 0.65];eE=ec;ep=0.6;
p2=p0*exp((interp1(T,S0,T2)-S00)*ec/R);
p4=p2*ep;
T5=interp1(S0,T,S04-log(p4/p0).* (ones(size(wcm)) *eE)*R);
h5=interp1(T,h,T5);
we=h4-h5;
wsp=we-wcm*ones(size(ec));
efficiency=wsp./((h4-h2)*ones(size(ec)));
plot(wsp,efficiency)
xlim([0 5e5])
ylim([0 1])
xlabel('w_s_p Specific Work (J kh^-^1)')
ylabel('Efficiency')
title(['OBC/MarsMix Performance Cross Plots, Hx Press. Ratio= '
num2str(ep) ',  Polytrope Efficiencies Listed']),
legend(num2str(ec'),'Location','SouthEast')
```



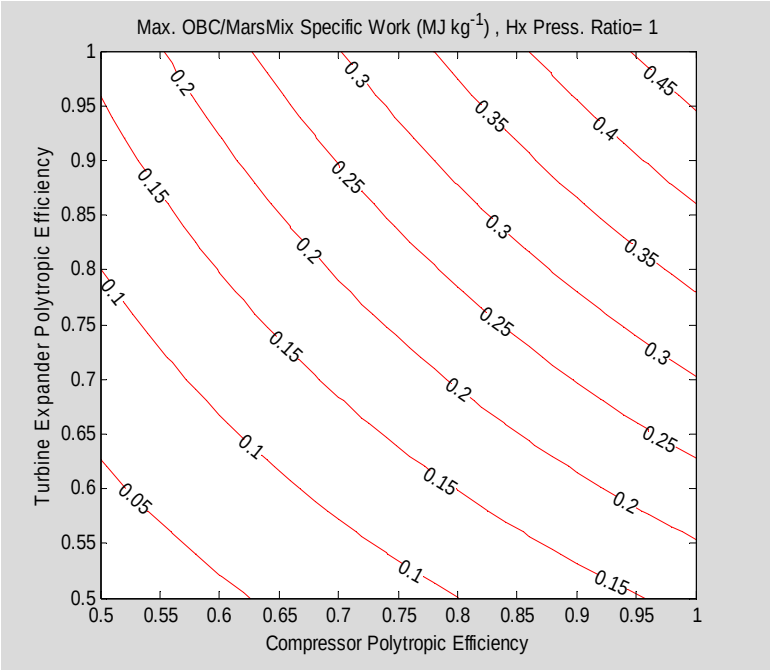

```
ec=[1 0.95 0.9 0.85 0.8 0.75 0.7 0.65];eE=ec;ep=0.4;
p2=p0*exp((interp1(T,S0,T2)-S00)*ec/R);
p4=p2*ep;
T5=interp1(S0,T,S04-log(p4/p0).* (ones(size(wcm)) *eE)*R);
h5=interp1(T,h,T5);
we=h4-h5;
wsp=we-wcm*ones(size(ec));
efficiency=wsp./((h4-h2)*ones(size(ec)));
plot(wsp,efficiency)
xlim([0 5e5])
ylim([0 1])
xlabel('w_s_p Specific Work (J kh^-^1)')
ylabel('Efficiency')
title(['OBC/MarsMix Performance Cross Plots, Hx Press. Ratio= '
num2str(ep) ',  Polytrope Efficiencies Listed']),
legend(num2str(ec'),'Location','SouthEast')
```



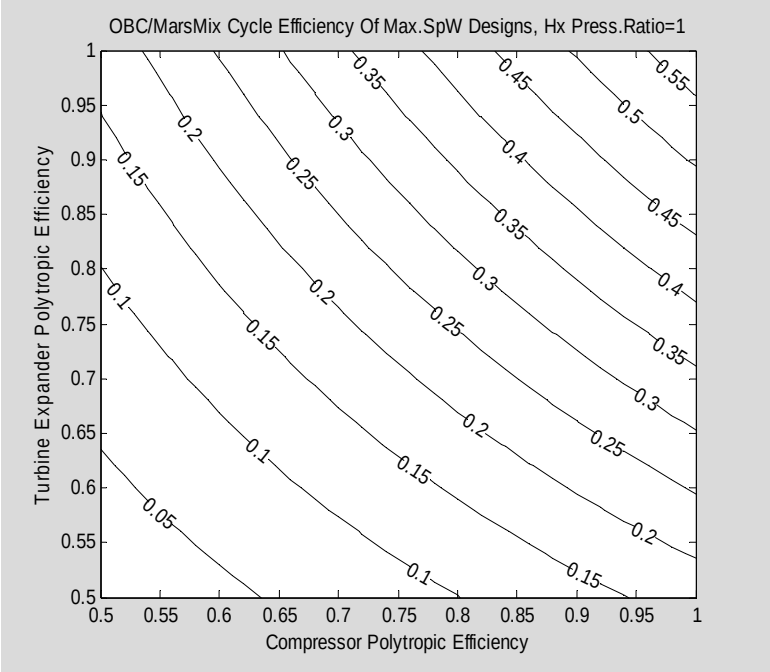
Next, generate contour plots showing how the maximum specific work optimized design point varies as a function of loss parameters.

```
clear;
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);
R=GasConstant('MarsMix');
T0=215;T4=1290;p0=850;
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);
wcm=linspace(0,1.4e6,1000)';
h2=h0+wcm;
T2=interp1(h,T,h2);

ep=1.0;
ec_vals=0.50:0.02:1.00;
eE_vals=0.50:0.02:1.00;
[Ec, EE]=meshgrid(ec_vals,eE_vals);
W_star=zeros(size(Ec));
Eff_star=W_star;
PIC_star=W_star;

for i=1:length(ec_vals);
for j=1:length(eE_vals);

ec=ec_vals(i);
eE=eE_vals(j);p2=p0*exp((interp1(T,S0,T2)-S00)*ec/R);
```

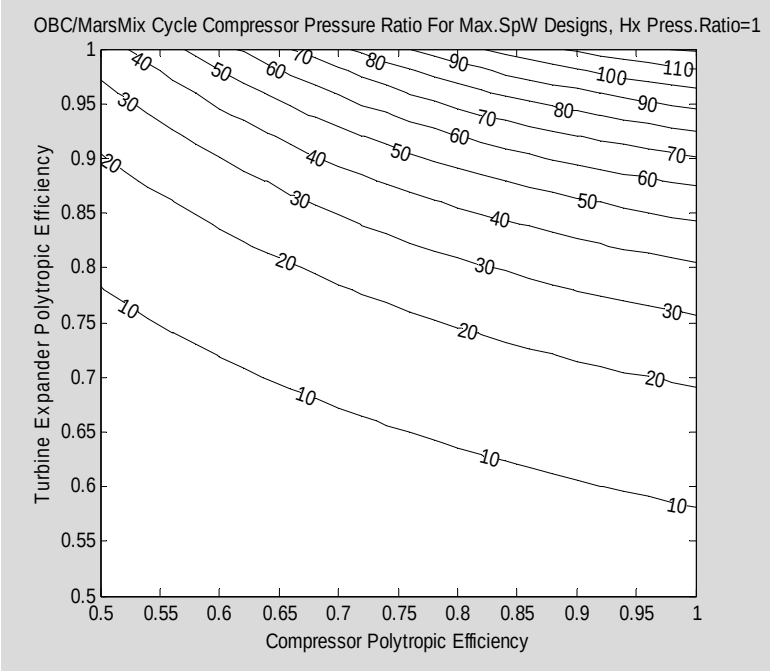
```
pic=p2/p0;
p4=p2*ep;
T5=interp1(S0,T,S04-log(p4/p0).* (ones(size(wcm)) *eE)*R);
h5=interp1(T,h,T5);
we=h4-h5;
wsp=we-wcm*ones(size(ec));
efficiency=wsp./((h4-h2)*ones(size(ec)));
I=find(wsp==max(wsp));
PIC_star(i,j)=pic(I);
W_star(i,j)=wsp(I);
Eff_star(i,j)=efficiency(I);
end
end
[C, handle]=contour(Ec,EE, W_star/1e6,0.05:0.05:0.5,'r');
clabel(C,handle);
title(['Max. OBC/MarsMix Specific Work (MJ kg^-^1) , Hx Press. Ratio= '
num2str(ep)])
xlabel('Compressor Polytropic Efficiency')
ylabel('Turbine Expander Polytropic Efficiency')
```



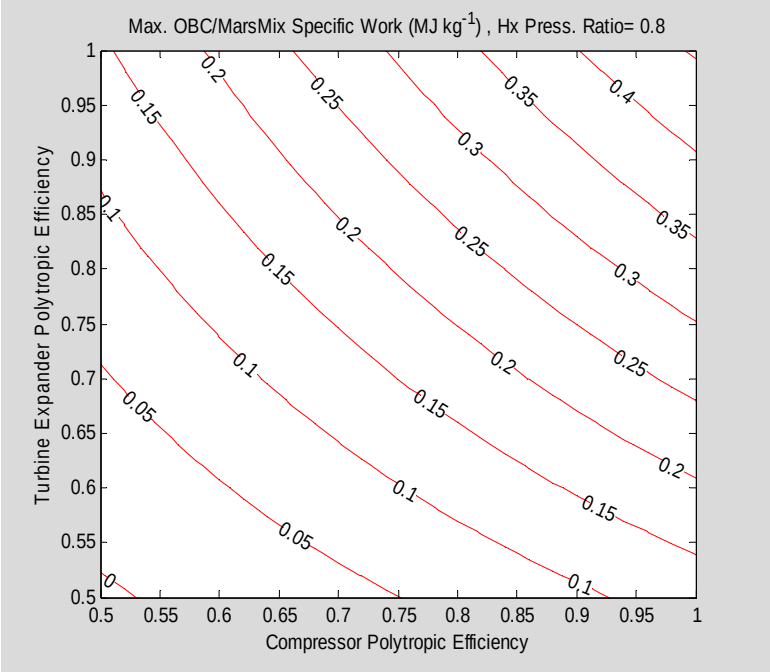

```
[C, handle]=contour(Ec,EE, Eff_star,'k');
clabel(C,handle);
title(['OBC/MarsMix Cycle Efficiency Of Max.SpW Designs, Hx
Press.Ratio=' num2str(ep)])
xlabel('Compressor Polytropic Efficiency')
ylabel('Turbine Expander Polytropic Efficiency')
```



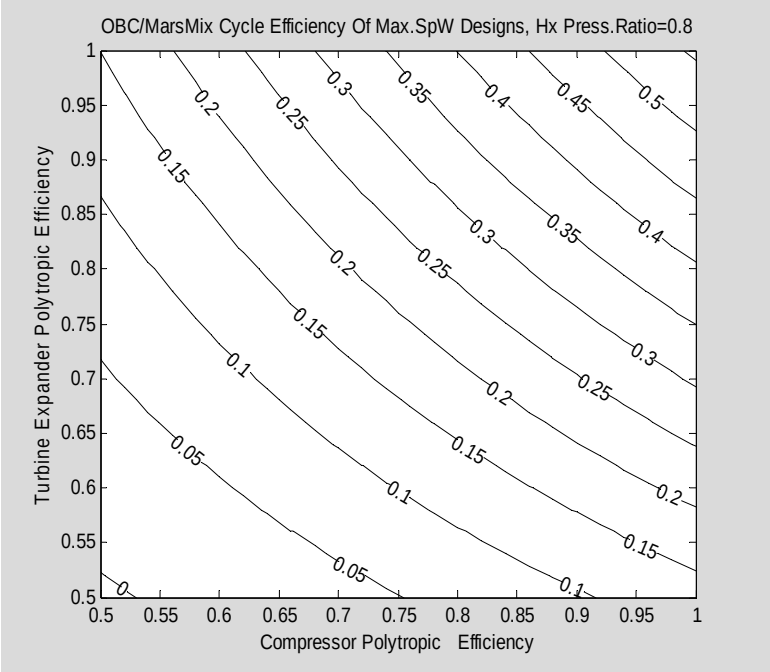

```
[C, handle]=contour(Ec,EE, PIC_star,10:10:130,'k');
clabel(C,handle);
title(['OBC/MarsMix Cycle Compressor Pressure Ratio For Max.SpW
Designs, Hx Press.Ratio=' num2str(ep)])
xlabel('Compressor Polytropic Efficiency')
ylabel('Turbine Expander Polytropic Efficiency')
```



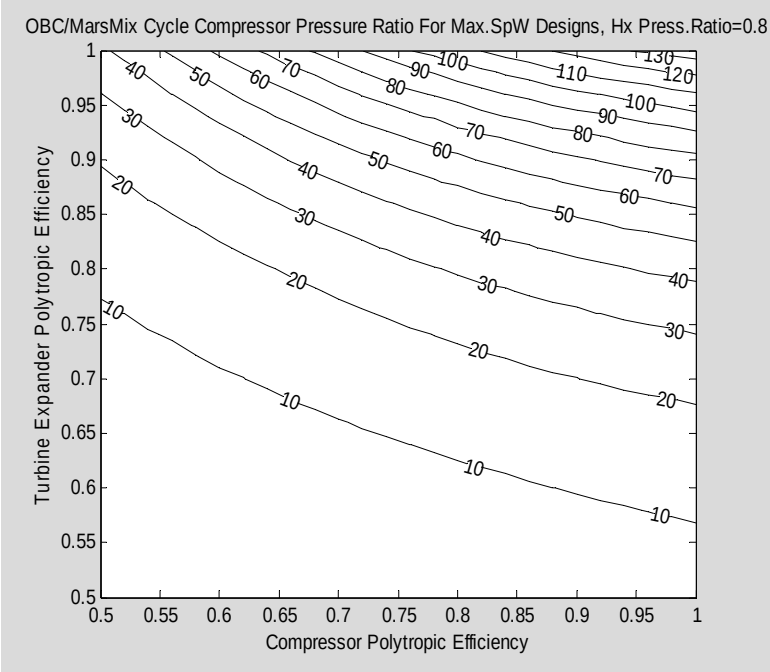
```
clear;
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);
R=GasConstant('MarsMix');
T0=215;T4=1290;p0=850;
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);
wcm=linspace(0,1.4e6,1000)';
h2=h0+wcm;
T2=interp1(h,T,h2);

ep=0.8;
ec_vals=0.50:0.02:1.00;
eE_vals=0.50:0.02:1.00;
[Ec, EE]=meshgrid(ec_vals,eE_vals);
W_star=zeros(size(Ec));
Eff_star=W_star;
PIC_star=W_star;

for i=1:length(ec_vals);
for j=1:length(eE_vals);

ec=ec_vals(i);
eE=eE_vals(j);p2=p0*exp((interp1(T,S0,T2)-S00)*ec/R);
pic=p2/p0;
p4=p2*ep;
T5=interp1(S0,T,S04-log(p4/p0).* (ones(size(wcm)) *eE)*R);
h5=interp1(T,h,T5);
we=h4-h5;
wsp=we-wcm*ones(size(ec));
efficiency=wsp./((h4-h2)*ones(size(ec)));
I=find(wsp==max(wsp));
PIC_star(i,j)=pic(I);
W_star(i,j)=wsp(I);
Eff_star(i,j)=efficiency(I);
end
end
[C, handle]=contour(Ec,EE, W_star/1e6,0.0:0.05:0.5,'r');
clabel(C,handle);
title(['Max. OBC/MarsMix Specific Work (MJ kg^-^1) , Hx Press. Ratio= '
num2str(ep)])
xlabel('Compressor Polytropic Efficiency')
ylabel('Turbine Expander Polytropic Efficiency')
```




```
[C, handle]=contour(Ec,EE, Eff_star,'k');
clabel(C,handle);
title(['OBC/MarsMix Cycle Efficiency Of Max.SpW Designs, Hx
Press.Ratio=' num2str(ep)])
xlabel('Compressor Polytropic   Efficiency')

ylabel('Turbine Expander Polytropic Efficiency')
```



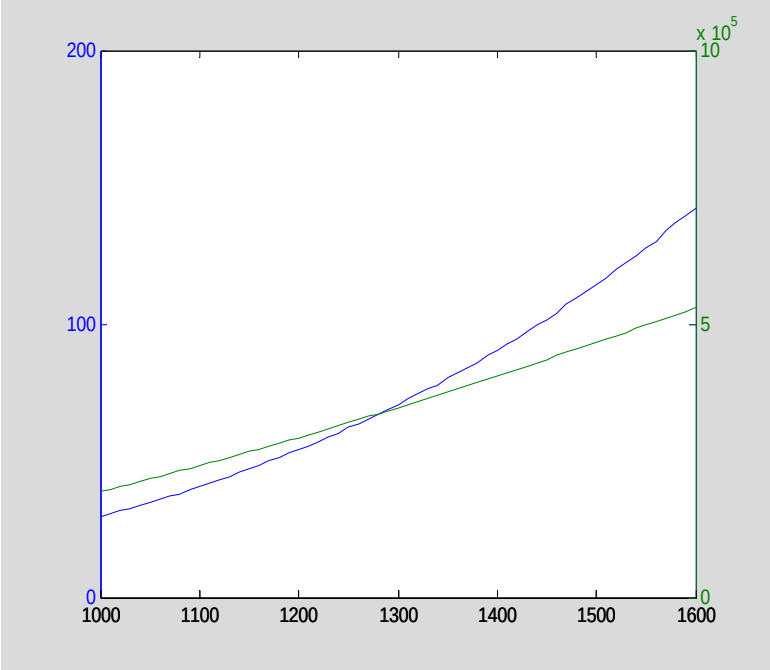

```
[C, handle]=contour(Ec,EE, PIC_star,10:10:130,'k');
clabel(C,handle);
title(['OBC/MarsMix Cycle Compressor Pressure Ratio For Max.SpW
Designs, Hx Press.Ratio=' num2str(ep)])
xlabel('Compressor Polytropic Efficiency')
ylabel('Turbine Expander Polytropic Efficiency')
```



Next, determine how cycle efficiency with lossy components is effected by chaning the maximum cycle temperature, T4.

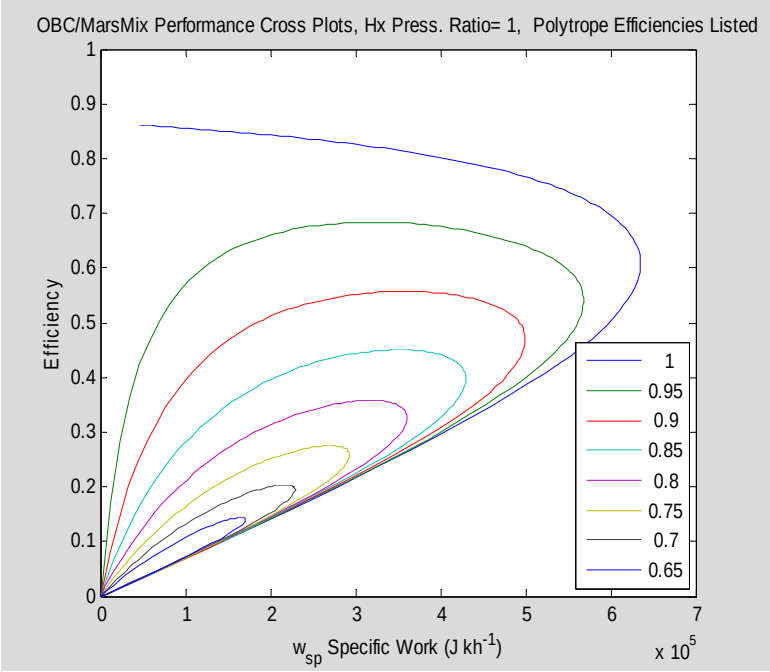
```
clear;
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);
R=GasConstant('MarsMix');
T0=215;p0=850;
Thot=1000:10:1600;
PIC=zeros(size(Thot));
WSP=PIC;
EFF=PIC;
for k=1:length(Thot);
T4=Thot(k);
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);
wcm=linspace(0,1.4e6,1000)';
h2=h0+wcm;
T2=interp1(h,T,h2);

ep=0.8;

ec=0.9;
eE=0.9;
p2=p0*exp((interp1(T,S0,T2)-S00)*ec/R);
pic=p2/p0;
p4=p2*ep;
```

```
T5=interp1(S0,T,S04-log(p4/p0).* (ones(size(wcm)) *eE)*R);
h5=interp1(T,h,T5);
we=h4-h5;
wsp=we-wcm*ones(size(ec));
efficiency=wsp./((h4-h2)*ones(size(ec)));
I=find(wsp==max(wsp));
PIC(k)=pic(I);
WSP(k)=wsp(I);
EFF(k)=efficiency(I);
end
plotyy(Thot,PIC,Thot,WSP)
```



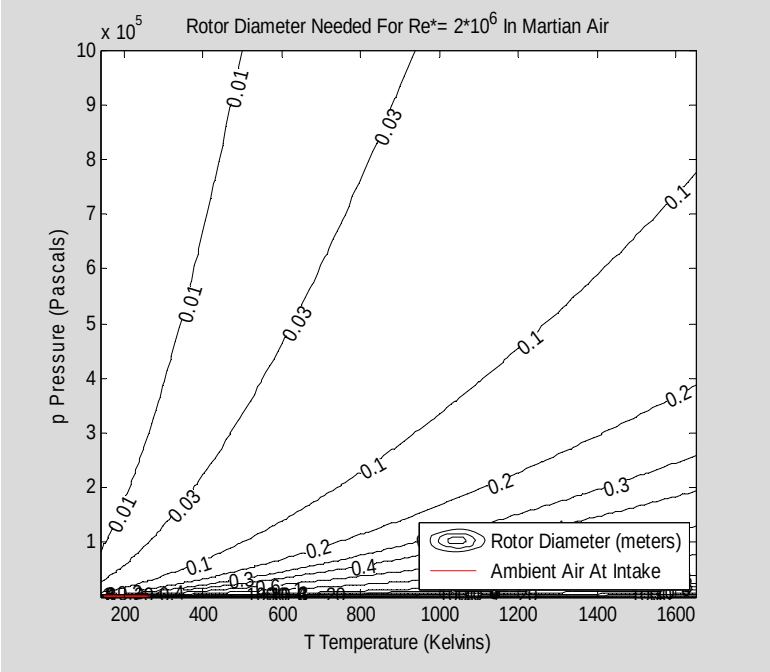

```
clear;
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);R=Ga
sConstant('MarsMix');
T0=215;T4=1505;p0=850;
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);
wcm=linspace(0,1.4e6)';
h2=h0+wcm;
T2=interp1(h,T,h2);

ec=[1 0.95 0.9 0.85 0.8 0.75 0.7 0.65];eE=ec;ep=1.0;
p2=p0*exp((interp1(T,S0,T2)-S00)*ec/R);
p4=p2*ep;
T5=interp1(S0,T,S04-log(p4/p0).* (ones(size(wcm)) *eE)*R);
h5=interp1(T,h,T5);
we=h4-h5;
```

```
wsp=we-wcm*ones(size(ec));
efficiency=wsp./((h4-h2)*ones(size(ec)));
plot(wsp,efficiency)
xlim([0 7e5])
ylim([0 1])
xlabel('w_s_p Specific Work (J kh^-^1)')
ylabel('Efficiency')
title(['OBC/MarsMix Performance Cross Plots, Hx Press. Ratio= '
num2str(ep) ',  Polytrope Efficiencies Listed']),
legend(num2str(ec'),'Location','SouthEast')
```



```
clear
vtip=450;
Re=2e6;
T=(140:10:1650)';
p=logspace(2,6,100);
[Tmesh,pmesh]=meshgrid(T,p);
R=GasConstant('MarsMix');
mu=Viscosity('MarsMix',T);
D=Re/vtip*(mu*ones(size(p)))'./(pmesh./Tmesh/R);
[C,h]=contour(Tmesh,pmesh,D,[100 30 20 10 6 4 3 2 1 0.6 0.4 0.3 0.2 0.1
0.03 0.01],'-k');
clabel(C,h);
hold on
T_ambient=[150 280 280 150 150];
```

```
p_ambient=[350 350 2000 2000 350];
p_start=linspace(p_ambient(1),p_ambient(2));
T_start=linspace(T_ambient(1),T_ambient(2));
for i=2:4
p_start=[p_start  linspace(p_ambient(i),p_ambient(i+1))];
T_start=[T_start  linspace(T_ambient(i),T_ambient(i+1))];
end
plot(T_start,p_start,'--r')
hold off
title('Rotor Diameter Needed For Re*= 2*10^6 In Martian Air')
xlabel('T Temperature (Kelvins)')
ylabel('p Pressure (Pascals)')
legend('Rotor Diameter (meters)','Ambient Air At
Intake','Location','SouthEast')
```



Next, evaluate

$$\left(\frac{p_{out}}{p_{in}}\right)_{compressor} = \exp\left(\frac{S_0\left[h^{-1}\left[E+h(T_0)\right]\right]-S_0\left[T_0\right]}{R}\right)$$

```
E=1.353E5;

p3E_in=1.8e5


clear;
```

```
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);
R=GasConstant('MarsMix');
T0=215;T4=1290;p0=850;
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);

mdot=1.0

ep=0.9

phic=0.0333;
psic=0.167;

vtipc1=500
vtipc2=500
vtipc3=475

vtipE3=475
vtipE2=500
vtipE1=500

Q1c=mdot*R*T0/p0
D1c=Q1c/phic/2/vtipc1
N1c=2*vtipc1/D1c

etac1=0.75
Ec1=psic*N1c^2*D1c^2
T1c_out_s=interp1(h,T,Ec1+h0)
T1c_out=interp1(h,T,Ec1/etac1+h0)

pic1=exp((interp1(T,S0,T1c_out_s)-S00)/R)
p1c_out=p0*pic1

p2c_in=p1c_out;

T2c_in=T1c_out;
Q2c=mdot*R*T2c_in/p2c_in
D2c=sqrt(Q2c/phic/2/vtipc2)
N2c=2*vtipc2/D2c
Ec2= psic*N2c^2*D2c^2
etac2=0.85
T2c_out_s=interp1(h,T,Ec2+Ec1/etac1+h0)
T2c_out=interp1(h,T,Ec2/etac2+Ec1/etac1+h0)

pic2=exp((interp1(T,S0,T2c_out_s)-interp1(T,S0,T2c_in))/R)
p2c_out=p2c_in*pic2

p3c_in=p2c_out;

T3c_in=T2c_out;
Q3c=mdot*R*T3c_in/p3c_in
D3c=sqrt(Q3c/phic/2/vtipc3)
N3c=2*vtipc3/D3c
```

```
Ec3= psic*N3c^2*D3c^2
etac3=0.85
T3c_out_s=interp1(h,T,Ec3+Ec2/etac2+Ec1/etac1+h0)
T3c_out=interp1(h,T,Ec3/etac3+Ec2/etac2+Ec1/etac1+h0)

phiE=0.0506
psiE=0.304

pic3=exp((interp1(T,S0,T3c_out_s)-interp1(T,S0,T3c_in))/R)
p3c_out=p3c_in*pic2

p3E_in=ep*p3c_out


T3E_in=T4
EE3=psiE*4*vtipE3.^2
etaE3=0.9
T3E_out_s=interp1(h,T,h4-EE3);
T3E_out=interp1(h,T,h4-etaE3*EE3)
piE3=exp((S04-interp1(T,S0,T3E_out_s))/R)
p3E_out=p3E_in./piE3
Q3E=mdot*R*T3E_out./p3E_out
D3E=sqrt(Q3E/phiE/2./vtipE3)
N3E=2*vtipE3./D3E



p2E_in=p3E_out
T2E_in=T3E_out
EE2=psiE*4*vtipE2^2
etaE2=0.9
T2E_out_s=interp1(h,T,h4-etaE3*EE3-EE2);
T2E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2)
piE2=exp((interp1(T,S0,T3E_out)-interp1(T,S0,T2E_out_s))/R)
p2E_out=p2E_in/piE2
Q2E=mdot*R*T2E_out/p2E_out
D2E=sqrt(Q2E/phiE/2/vtipE2)
N2E=2*vtipE2/D2E

p1E_in=p2E_out
T1E_in=T2E_out
EE1=psiE*4*vtipE1^2
etaE1=0.75
T1E_out_s=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-EE1);
T1E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-etaE1*EE1)
piE1=exp((interp1(T,S0,T2E_out)-interp1(T,S0,T1E_out_s))/R)
p1E_out=p1E_in/piE1
Q1E=mdot*R*T1E_out/p1E_out
D1E=sqrt(Q1E/phiE/2/vtipE1)
N1E=2*vtipE1/D1E
[N1c N1E]
[N2c N2E]
[N3c N3E]
```

```
mdot =
     1
ep =
    0.9000
vtipc1 =
   500
vtipc2 =
   500
vtipc3 =
   475
vtipE3 =
   475
vtipE2 =
   500
vtipE1 =
   500
Q1c =
   48.3270
D1c =
    1.4513
N1c =
  689.0553
etac1 =
    0.7500
Ec1 =
      167000
T1c_out_s =
  410.7706
T1c_out =
  468.8237
pic1 =
   19.1871
p1c_out =
  1.6309e+004
Q2c =
    5.4923
D2c =
    0.4061
N2c =
  2.4623e+003
Ec2 =
      167000
etac2 =
    0.8500
T2c_out_s =
  631.1655
T2c_out =
  658.4168
pic2 =
    4.7898
p2c_out =
  7.8117e+004
Q3c =
    1.6104
```

```
D3c =
    0.2256
N3c =
  4.2106e+003
Ec3 =
  1.5072e+005
etac3 =
    0.8500
T3c_out_s =
  793.0184
T3c_out =
  816.0801
phiE =
    0.0506
psiE =
    0.3040
pic3 =
    2.9996
p3c_out =
  3.7416e+005
p3E_in =
  3.3675e+005
T3E_in =
        1290
EE3 =
      274360
etaE3 =
    0.9000
T3E_out =
  1.0933e+003
piE3 =
    3.3854
p3E_out =
  9.9470e+004
Q3E =
    2.1000
D3E =
    0.2090
N3E =
  4.5451e+003
p2E_in =
  9.9470e+004
T2E_in =
  1.0933e+003
EE2 =
      304000
etaE2 =
    0.9000
T2E_out =
  866.9027
piE2 =
    5.2841
p2E_out =
  1.8824e+004
Q2E =
```

```
    8.7988
D2E =
    0.4170
N2E =
  2.3981e+003
p1E_in =
  1.8824e+004
T1E_in =
  866.9027
EE1 =
      304000
etaE1 =
    0.7500
T1E_out =
  666.3303
piE1 =
    9.1026
p1E_out =
  2.0680e+003
Q1E =
   61.5617
D1E =
    1.1030
N1E =
  906.6091
ans =
  689.0553  906.6091
ans =
  1.0e+003 *
    2.4623    2.3981
ans =
  1.0e+003 *
    4.2106    4.5451


-------------------------------------
```

Engine Model Number	Output Shaft Power Design Rating	
	horsepower	kilowatts
1	100	74.6
2	178	132.7
3	316	235.9
4	562	419.5
5	1,000	746.0
6	1,780	1,326.6
7	3,160	2,359.0
8	5,620	4,195.1
9	10,000	7,460.0

First we initialize variables in preparation for an iterative set of calculations.

```
clear
f_eta_1c=1;
f_eta_2c=1;
f_eta_3c=1;
f_eta_3E=1;
f_eta_2E=1;
f_eta_1E=1;
EngineModel=1
hp=[100 176 316 562 1000 1780 3160 5620 10000];
ep=0.9
```

The following MATLAB cell needs to be executed several times (e.g., perhaps 5) until its calculated quantities stop changing. Only then are its set of calculated quantities self-consistent. As implemented here, the user must manually initiate each of those iterative executions and decide when to stop iterating. .

```
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);
R=GasConstant('MarsMix');
T0=215;T4=1290;p0=850;
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);

Ec1=1.27e5
Ec2=1.27e5
Ec3=1.27e5
EE3=2.5e5
EE2=2.95e5
EE1=2.95e5
display(' ')
etac1=0.85*f_eta_1c
etac2=0.85*f_eta_2c
etac3=0.85*f_eta_3c
etaE3=0.88*f_eta_3E
etaE2=0.88*f_eta_2E
etaE1=0.88*f_eta_1E
display(' ')
T1c_out_s=interp1(h,T,Ec1+h0);
T1c_out=interp1(h,T,Ec1/etac1+h0);
pic1=exp((interp1(T,S0,T1c_out_s)-S00)/R)
p1c_out=p0*pic1;
p2c_in=p1c_out;
T2c_in=T1c_out;
T2c_out_s=interp1(h,T,Ec2+Ec1/etac1+h0);
T2c_out=interp1(h,T,Ec2/etac2+Ec1/etac1+h0);
pic2=exp((interp1(T,S0,T2c_out_s)-interp1(T,S0,T2c_in))/R)
p2c_out=p2c_in*pic2
p3c_in=p2c_out;
T3c_in=T2c_out;
T3c_out_s=interp1(h,T,Ec3+Ec2/etac2+Ec1/etac1+h0);
T3c_out=interp1(h,T,Ec3/etac3+Ec2/etac2+Ec1/etac1+h0);
pic3=exp((interp1(T,S0,T3c_out_s)-interp1(T,S0,T3c_in))/R)
```

```
p3c_out=p3c_in*pic3;
p3E_in=ep*p3c_out;
T3E_in=T4;
T3E_out_s=interp1(h,T,h4-EE3);;
T3E_out=interp1(h,T,h4-etaE3*EE3);
piE3=exp((S04-interp1(T,S0,T3E_out_s))/R);
p3E_out=p3E_in./piE3
p2E_in=p3E_out;
T2E_in=T3E_out;
T2E_out_s=interp1(h,T,h4-etaE3*EE3-EE2);
T2E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2);
piE2=exp((interp1(T,S0,T3E_out)-interp1(T,S0,T2E_out_s))/R);
p2E_out=p2E_in/piE2
p1E_in=p2E_out
T1E_in=T2E_out;
T1E_out_s=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-EE1);
T1E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-etaE1*EE1);
piE1=exp((interp1(T,S0,T2E_out)-interp1(T,S0,T1E_out_s))/R)
p1E_out=p1E_in/piE1;
display(' ')
Wsp=EE1*etaE1+EE2*etaE2+EE3*etaE3-Ec1/etac1-Ec2/etac2-Ec3/etac3
HeatAdded=h4-(Ec3+Ec2/etac2+Ec1/etac1+h0)
CycleEfficiency=Wsp/HeatAdded
mdot=746*hp(EngineModel)/Wsp
ReactorPower=HeatAdded*mdot
display(' ')
ShaftPower1c=-Ec1/etac1*mdot
ShaftPower2c=-Ec2/etac2*mdot
ShaftPower3c=-Ec3/etac3*mdot
ShaftPower3E=+EE3*etaE3*mdot
ShaftPower2E=+EE2*etaE2*mdot
ShaftPower1E=+EE1*etaE1*mdot
display(' ')
Q1c=mdot*R*T0/p0
Q2c=mdot*R*T2c_in/p2c_in
Q3c=mdot*R*T3c_in/p3c_in
Q3E=mdot*R*T3E_out./p3E_out
Q2E=mdot*R*T2E_out/p2E_out
Q1E=mdot*R*T1E_out/p1E_out
display(' ')
display('                    Pressures   Temperatures')
display([p0 T0; p2c_in T2c_in; p3c_in T3c_in; p3c_out T3c_out; p3E_in
T3E_in; p2E_in T2E_in; p1E_in T1E_in; p1E_out T1E_out])
display(' ')
NsE_ideal=0.55, DsE_ideal=3.3
Nsc_ideal=0.70,Dsc_ideal=3.7
D1c_ideal=Dsc_ideal*Q1c^0.50/Ec1^0.25;
D2c_ideal=Dsc_ideal*Q2c^0.50/Ec2^0.25;
D3c_ideal=Dsc_ideal*Q3c^0.50/Ec3^0.25;
D3E_ideal=DsE_ideal*Q3E^0.50/EE3^0.25;
D2E_ideal=DsE_ideal*Q2E^0.50/EE2^0.25;
D1E_ideal=DsE_ideal*Q1E^0.50/EE1^0.25;
display(' ')
N1c_ideal=Nsc_ideal*Q1c^-0.50*Ec1^0.75;
N2c_ideal=Nsc_ideal*Q2c^-0.50*Ec2^0.75;
```

```
N3c_ideal=Nsc_ideal*Q3c^-0.50*Ec3^0.75;
N3E_ideal=NsE_ideal*Q3E^-0.50*EE3^0.75;
N2E_ideal=NsE_ideal*Q2E^-0.50*EE2^0.75;
N1E_ideal=NsE_ideal*Q1E^-0.50*EE1^0.75;
display(' ')
N1=(N1c_ideal+N1E_ideal)/2
N2=(N2c_ideal+N2E_ideal)/2
N3=(N3c_ideal+N3E_ideal)/2
display(' ')
N1rpm=N1*30/pi
N2rpm=N2*30/pi
N3rpm=N3*30/pi
display(' ')
Ns1c=N1*Q1c^0.5*Ec1^-0.75
Ns2c=N2*Q2c^0.5*Ec2^-0.75
Ns3c=N3*Q3c^0.5*Ec3^-0.75
Ns3E=N3*Q3E^0.5*EE3^-0.75
Ns2E=N2*Q2E^0.5*EE2^-0.75
Ns1E=N1*Q1E^0.5*EE1^-0.75
display(' ')
Ds1c= 10^(log10(Dsc_ideal)-0.5*log10(Ns1c/Nsc_ideal))
Ds2c= 10^(log10(Dsc_ideal)-0.5*log10(Ns2c/Nsc_ideal))
Ds3c= 10^(log10(Dsc_ideal)-0.5*log10(Ns3c/Nsc_ideal))
Ds3E= 10^(log10(DsE_ideal)-1.0*log10(Ns3E/NsE_ideal))
Ds2E= 10^(log10(DsE_ideal)-1.0*log10(Ns2E/NsE_ideal))
Ds1E= 10^(log10(DsE_ideal)-1.0*log10(Ns1E/NsE_ideal))
display(' ')
D1c=Ds1c*Q1c^0.50/Ec1^0.25
D2c=Ds2c*Q2c^0.50/Ec2^0.25
D3c=Ds3c*Q3c^0.50/Ec3^0.25
D3E=Ds3E*Q3E^0.50/EE3^0.25
D2E=Ds2E*Q2E^0.50/EE2^0.25
D1E=Ds1E*Q1E^0.50/EE1^0.25
display(' ')
vtip1c=N1*D1c/2
vtip2c=N2*D2c/2
vtip3c=N3*D3c/2
vtip3E=N3*D3E/2
vtip2E=N2*D2E/2
vtip1E=N1*D1E/2
display(' ')
Re1c=vtip1c*D1c*p0/R/T0/Viscosity('MarsMix',T0)
Re2c=vtip2c*D2c*p2c_in/R/T2c_in/ Viscosity('MarsMix',T2c_in)
Re3c=vtip3c*D3c*p3c_in/R/T3c_in/ Viscosity('MarsMix',T3c_in)
Re3E=vtip3E*D3E*p3E_out/R/T3E_out/ Viscosity('MarsMix',T3E_out)
Re2E=vtip2E*D2E*p2E_out/R/T2E_out/ Viscosity('MarsMix',T2E_out)
Re1E=vtip1E*D1E*p1E_out/R/T1E_out/ Viscosity('MarsMix',T1E_out)
log10Re_val=[3 3.5 4 4.5 5 5.5 6 6.5 7];
f_eta_val=[0.2 0.34 0.5 0.65 0.8 0.915 0.97 0.99 1.];
display(' ')
f_eta_1c=interp1(log10Re_val,f_eta_val,log10(Re1c),'linear','extrap' )
f_eta_2c=interp1(log10Re_val,f_eta_val,log10(Re2c),'linear','extrap' )
f_eta_3c=interp1(log10Re_val,f_eta_val,log10(Re3c),'linear','extrap' )
f_eta_3E=interp1(log10Re_val,f_eta_val,log10(Re3E),'linear','extrap' )
f_eta_2E=interp1(log10Re_val,f_eta_val,log10(Re2E),'linear','extrap' )
```

```
f_eta_1E=interp1(log10Re_val,f_eta_val,log10(Re1E),'linear','extrap' )


Ec1 =
      127000
Ec2 =
      127000
Ec3 =
      127000
EE3 =
      250000
EE2 =
      295000
EE1 =
      295000

etac1 =
         0.813418162884625
etac2 =
          0.82537638376349
etac3 =
         0.829222759759394
etaE3 =
         0.811995656412047
etaE2 =
         0.772401426252376
etaE1 =
         0.712464511090386

pic1 =
          11.2851768923859
pic2 =
          4.01409589963562
p2c_out =
          38504.8149468305
pic3 =
          2.94366792160736
p3E_out =
          33924.6432411538
p2E_out =
          7210.89351028668
p1E_in =
          7210.89351028668
piE1 =
          6.85117149141829

Wsp =
          177878.445414266
HeatAdded =
           739289.02614919
CycleEfficiency =
          0.24060744732111
mdot =
         0.419387519529205
```

```
ReactorPower =
           310048.59089187

ShaftPower1c =
         -65479.5004715966
ShaftPower2c =
         -64530.8201542525
ShaftPower3c =
         -64231.4919041338
ShaftPower3E =
           85135.2110527842
ShaftPower2E =
           95560.9778798572
ShaftPower1E =
           88145.6235973415

Q1c =
           20.2677552361717
Q2c =
           3.33400536732201
Q3c =
           1.15615857857776
Q3E =
           2.66606766511993
Q2E =
           10.4717097590455
Q1E =
           58.0680817112111

                    Pressures   Temperatures
ans =
                         850                        215
           9592.40035852798          399.123661045476
           38504.8149468305          555.579649935401
           113345.388586412          698.500040760012
           102010.849727771                      1290
           33924.6432411538            1128.756697438
           7210.89351028668          942.368068281595
           1052.50518386804          762.738390629831

NsE_ideal =
                       0.55
DsE_ideal =
                        3.3
Nsc_ideal =
                        0.7
Dsc_ideal =
                        3.7


N1 =
           979.824252138242
N2 =
           2365.24663605694
N3 =
```

```
          4072.84651692852

N1rpm =
          9356.63238534725
N2rpm =
          22586.4416255963
N3rpm =
           38892.819337427

Ns1c =
         0.655689485441578
Ns2c =
         0.641958258299027
Ns3c =
         0.650959480645231
Ns3E =
         0.594810493404788
Ns2E =
         0.604670414396772
Ns1E =
         0.589861990954527

Ds1c =
           3.8229765524619
Ds2c =
          3.86364618642154
Ds3c =
          3.83684067812763
Ds3E =
          3.05139203178925
Ds2E =
          3.00163519958335
Ds1E =
          3.07699093657981

D1c =
         0.911703305858117
D2c =
         0.373705565827669
D3c =
         0.218540301835192
D3E =
         0.222817136915922
D2E =
         0.416784158805453
D1E =
          1.00609607021376

vtip1c =
          446.654504917197
vtip2c =
          441.952916224824
vtip3c =
          445.040553568984
vtip3E =
```

```
                   453.75
vtip2E =
            492.89866478821
vtip1E =
            492.89866478821

Re1c =
           761184.656322237
Re2c =
           1061172.74971023
Re3c =
           1376937.66248951
Re3E =
           371717.653130415
Re2E =
           217750.956392792
Re1E =
            110110.50720303

f_eta_1c =
          0.956963902789188
f_eta_2c =
          0.971031443561103
f_eta_3c =
          0.975556571162938
f_eta_3E =
          0.922723450518643
f_eta_2E =
          0.877730816368242
f_eta_1E =
          0.809620615521457
```

Results Extracted From MATLAB Output:

Summary of Engine Model #1 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**100 horsepower**					
Output Power (SI units)	**74.6 kW**					
Thermal Input Power (from Nuclear Reactor)	**310.0 kW**					
Thermal Conversion Cycle Energy Efficiency	**0.2406**					
Air Mass Flow Rate (kg/sec)	**0.4194 kg/sec**					
	Low Pressure Stages		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**979.8 radians/sec**		**2,365.2 radians/sec**		**4,072.8 radians/sec**	
Shaft Speed (rpm)	**9,356.6 rpm**		**22,586.4 rpm**		**38,892.8 rpm**	
	Compressor	**Turbine**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**0.912 m**	**1.006 m**	**0.374 m**	**0.417**	**0.219 m**	**0.223 m**
Rotor Tip Speed (m/s)	**446.7 m/s**	**492.9 m/s**	**442.0 m/s**	**492.9 m/s**	**445.0 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.07**	**3.86**	**3.00**	**3.84**	**3.05**
Ns: Specific Speed	**0.656**	**0.590**	**0.642**	**0.605**	**0.651**	**0.595**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$7.68*10^5$**	**$1.10*10^5$**	**$1.06*10^6$**	**$2.18*10^5$**	**$1.38*10^6$**	**$3.71*10^5$**
Reynolds Number Efficiency Factor	**0.957**	**0.810**	**0.971**	**0.878**	**0.976**	**0.923**
Assumed Efficiency (from Balje*Reynolds)	**0.813**	**0.712**	**0.825**	**0.772**	**0.829**	**0.812**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**942.49 Kelvins**	**399.1 Kelvins**	**1128.8 Kelvins**	**555.6 Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,211 Pa**	**9,592 Pa**	**33,924 Pa**	**38,504 Pa**	**102,011 Pa**
Pressure Ratio (Calculated)	**11.285**	**6.851**	**4.014**	**4.705**	**2.944**	**3.007**
Discharge Air Temperature (Kelvins)	**399.1 Kelvins**	**762.7 Kelvins**	**555.6 Kelvins**	**942.49 Kelvins**	**698.5 Kelvins**	**1128.8 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**1,053 Pa**	**38,504 Pa**	**7,211 Pa**	**113,345 Pa**	**33,924 Pa**
Q:Volumetric Flow Rate On Low Pressure Side	**20.27 m^3/s**	**58.07 m^3/s**	**3.33 m^3/s**	**10.47 m^3/s**	**1.16 m^3/s**	**2.67 m^3/s**
Shaft Powers (kW) [Total=+74.6 kW=100 hp]	**-65.4795 kW**	**+88.1456 kW**	**-64.5308 kW**	**+95.5610 kW**	**-64.2315**	**+85.1352 kW**

First we initialize variables in preparation for an iterative set of calculations.

```
clear
f_eta_1c=1;
f_eta_2c=1;
f_eta_3c=1;
f_eta_3E=1;
f_eta_2E=1;
f_eta_1E=1;
EngineModel=2
hp=[100 176 316 562 1000 1780 3160 5620 10000];
ep=0.9
```

```
EngineModel =
     2
ep =
                        0.9
```

The following MATLAB cell needs to be executed several times (e.g., perhaps 5) until its calculated quantities stop changing. Only then are its set of calculated quantities self-consistent. As implemented here, the user must manually initiate each of those iterative executions and decide when to stop iterating. .

```
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);
R=GasConstant('MarsMix');
T0=215;T4=1290;p0=850;
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);

Ec1=1.27e5
Ec2=1.27e5
Ec3=1.27e5
EE3=2.5e5
EE2=2.95e5
EE1=2.95e5
display(' ')
etac1=0.85*f_eta_1c
etac2=0.85*f_eta_2c
etac3=0.85*f_eta_3c
etaE3=0.88*f_eta_3E
etaE2=0.88*f_eta_2E
etaE1=0.88*f_eta_1E
display(' ')
T1c_out_s=interp1(h,T,Ec1+h0);
T1c_out=interp1(h,T,Ec1/etac1+h0);
pic1=exp((interp1(T,S0,T1c_out_s)-S00)/R)
p1c_out=p0*pic1;
p2c_in=p1c_out;
T2c_in=T1c_out;
T2c_out_s=interp1(h,T,Ec2+Ec1/etac1+h0);
T2c_out=interp1(h,T,Ec2/etac2+Ec1/etac1+h0);
```

```
pic2=exp((interp1(T,S0,T2c_out_s)-interp1(T,S0,T2c_in))/R)
p2c_out=p2c_in*pic2
p3c_in=p2c_out;
T3c_in=T2c_out;
T3c_out_s=interp1(h,T,Ec3+Ec2/etac2+Ec1/etac1+h0);
T3c_out=interp1(h,T,Ec3/etac3+Ec2/etac2+Ec1/etac1+h0);
pic3=exp((interp1(T,S0,T3c_out_s)-interp1(T,S0,T3c_in))/R)
p3c_out=p3c_in*pic3;
p3E_in=ep*p3c_out;
T3E_in=T4;
T3E_out_s=interp1(h,T,h4-EE3);;
T3E_out=interp1(h,T,h4-etaE3*EE3);
piE3=exp((S04-interp1(T,S0,T3E_out_s))/R)
p3E_out=p3E_in./piE3;
p2E_in=p3E_out;
T2E_in=T3E_out;
T2E_out_s=interp1(h,T,h4-etaE3*EE3-EE2);
T2E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2);
piE2=exp((interp1(T,S0,T3E_out)-interp1(T,S0,T2E_out_s))/R)
p2E_out=p2E_in/piE2;
p1E_in=p2E_out
T1E_in=T2E_out;
T1E_out_s=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-EE1);
T1E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-etaE1*EE1);
piE1=exp((interp1(T,S0,T2E_out)-interp1(T,S0,T1E_out_s))/R)
p1E_out=p1E_in/piE1;
display(' ')
Wsp=EE1*etaE1+EE2*etaE2+EE3*etaE3-Ec1/etac1-Ec2/etac2-Ec3/etac3
HeatAdded=h4-(Ec3+Ec2/etac2+Ec1/etac1+h0)
CycleEfficiency=Wsp/HeatAdded
mdot=746*hp(EngineModel)/Wsp
ReactorPower=HeatAdded*mdot
display(' ')
ShaftPower1c=-Ec1/etac1*mdot
ShaftPower2c=-Ec2/etac2*mdot
ShaftPower3c=-Ec3/etac3*mdot
ShaftPower3E=+EE3*etaE3*mdot
ShaftPower2E=+EE2*etaE2*mdot
ShaftPower1E=+EE1*etaE1*mdot
display(' ')
Q1c=mdot*R*T0/p0
Q2c=mdot*R*T2c_in/p2c_in
Q3c=mdot*R*T3c_in/p3c_in
Q3E=mdot*R*T3E_out./p3E_out
Q2E=mdot*R*T2E_out/p2E_out
Q1E=mdot*R*T1E_out/p1E_out
display(' ')
display('                    Pressures   Temperatures')
display([p0 T0; p2c_in T2c_in; p3c_in T3c_in; p3c_out T3c_out; p3E_in
T3E_in; p2E_in T2E_in; p1E_in T1E_in; p1E_out T1E_out])
display(' ')
NsE_ideal=0.55, DsE_ideal=3.3
Nsc_ideal=0.70,Dsc_ideal=3.7
D1c_ideal=Dsc_ideal*Q1c^0.50/Ec1^0.25;
D2c_ideal=Dsc_ideal*Q2c^0.50/Ec2^0.25;
```

```
D3c_ideal=Dsc_ideal*Q3c^0.50/Ec3^0.25;
D3E_ideal=DsE_ideal*Q3E^0.50/EE3^0.25;
D2E_ideal=DsE_ideal*Q2E^0.50/EE2^0.25;
D1E_ideal=DsE_ideal*Q1E^0.50/EE1^0.25;
display(' ')
N1c_ideal=Nsc_ideal*Q1c^-0.50*Ec1^0.75;
N2c_ideal=Nsc_ideal*Q2c^-0.50*Ec2^0.75;
N3c_ideal=Nsc_ideal*Q3c^-0.50*Ec3^0.75;
N3E_ideal=NsE_ideal*Q3E^-0.50*EE3^0.75;
N2E_ideal=NsE_ideal*Q2E^-0.50*EE2^0.75;
N1E_ideal=NsE_ideal*Q1E^-0.50*EE1^0.75;
display(' ')
N1=(N1c_ideal+N1E_ideal)/2
N2=(N2c_ideal+N2E_ideal)/2
N3=(N3c_ideal+N3E_ideal)/2
display(' ')
N1rpm=N1*30/pi
N2rpm=N2*30/pi
N3rpm=N3*30/pi
display(' ')
Ns1c=N1*Q1c^0.5*Ec1^-0.75
Ns2c=N2*Q2c^0.5*Ec2^-0.75
Ns3c=N3*Q3c^0.5*Ec3^-0.75
Ns3E=N3*Q3E^0.5*EE3^-0.75
Ns2E=N2*Q2E^0.5*EE2^-0.75
Ns1E=N1*Q1E^0.5*EE1^-0.75
display(' ')
Ds1c= 10^(log10(Dsc_ideal)-0.5*log10(Ns1c/Nsc_ideal))
Ds2c= 10^(log10(Dsc_ideal)-0.5*log10(Ns2c/Nsc_ideal))
Ds3c= 10^(log10(Dsc_ideal)-0.5*log10(Ns3c/Nsc_ideal))
Ds3E= 10^(log10(DsE_ideal)-1.0*log10(Ns3E/NsE_ideal))
Ds2E= 10^(log10(DsE_ideal)-1.0*log10(Ns2E/NsE_ideal))
Ds1E= 10^(log10(DsE_ideal)-1.0*log10(Ns1E/NsE_ideal))
display(' ')
D1c=Ds1c*Q1c^0.50/Ec1^0.25
D2c=Ds2c*Q2c^0.50/Ec2^0.25
D3c=Ds3c*Q3c^0.50/Ec3^0.25
D3E=Ds3E*Q3E^0.50/EE3^0.25
D2E=Ds2E*Q2E^0.50/EE2^0.25
D1E=Ds1E*Q1E^0.50/EE1^0.25
display(' ')
vtip1c=N1*D1c/2
vtip2c=N2*D2c/2
vtip3c=N3*D3c/2
vtip3E=N3*D3E/2
vtip2E=N2*D2E/2
vtip1E=N1*D1E/2
display(' ')
Re1c=vtip1c*D1c*p0/R/T0/Viscosity('MarsMix',T0)
Re2c=vtip2c*D2c*p2c_in/R/T2c_in/ Viscosity('MarsMix',T2c_in)
Re3c=vtip3c*D3c*p3c_in/R/T3c_in/ Viscosity('MarsMix',T3c_in)
Re3E=vtip3E*D3E*p3E_out/R/T3E_out/ Viscosity('MarsMix',T3E_out)
Re2E=vtip2E*D2E*p2E_out/R/T2E_out/ Viscosity('MarsMix',T2E_out)
Re1E=vtip1E*D1E*p1E_out/R/T1E_out/ Viscosity('MarsMix',T1E_out)
log10Re_val=[3 3.5 4 4.5 5 5.5 6 6.5 7];
```

```
f_eta_val=[0.2 0.34 0.5 0.65 0.8 0.915 0.97 0.99 1.];
display(' ')
f_eta_1c=interp1(log10Re_val,f_eta_val,log10(Re1c),'linear','extrap' )
f_eta_2c=interp1(log10Re_val,f_eta_val,log10(Re2c),'linear','extrap' )
f_eta_3c=interp1(log10Re_val,f_eta_val,log10(Re3c),'linear','extrap' )
f_eta_3E=interp1(log10Re_val,f_eta_val,log10(Re3E),'linear','extrap' )
f_eta_2E=interp1(log10Re_val,f_eta_val,log10(Re2E),'linear','extrap' )
f_eta_1E=interp1(log10Re_val,f_eta_val,log10(Re1E),'linear','extrap' )


Ec1 =
      127000
Ec2 =
      127000
Ec3 =
      127000
EE3 =
      250000
EE2 =
      295000
EE1 =
      295000

etac1 =
         0.822879779235004
etac2 =
         0.828913840534301
etac3 =
         0.832787235026905
etaE3 =
         0.822091581569519
etaE2 =
         0.794208257251742
etaE1 =
         0.734456923880708

pic1 =
          11.2851768923859
pic2 =
          4.03729383629516
p2c_out =
          38727.3388427606
pic3 =
          2.95437330622441
piE3 =
          3.00698371395171
piE2 =
          4.72024751082682
p1E_in =
          7254.87088415865
piE1 =
          6.98105024303579

Wsp =
          196430.609712623
```

```
HeatAdded =
          741740.900004056
CycleEfficiency =
         0.264823754105442
mdot =
         0.668409064107093
ReactorPower =
          495786.340781664

ShaftPower1c =
         -103159.602755724
ShaftPower2c =
         -102408.654543497
ShaftPower3c =
         -101932.339463463
ShaftPower3E =
          137373.366161801
ShaftPower2E =
           156602.51939105
ShaftPower1E =
          144820.711209833

Q1c =
           32.302228078148
Q2c =
          5.28789482698004
Q3c =
          1.82423457873995
Q3E =
          4.20181487250693
Q2E =
          16.4568211065078
Q1E =
          92.0558300696347

                  Pressures   Temperatures
ans =
                        850                        215
          9592.40035852798          397.188952377615
          38727.3388427606          553.204104794062
          114415.016098159          695.693364018211
          102973.514488343                      1290
          34244.7862323199          1126.72629144832
          7254.87088415865          934.894685011759
          1039.22341647606          749.112902567775

NsE_ideal =
                      0.55
DsE_ideal =
                       3.3
Nsc_ideal =
                       0.7
Dsc_ideal =
                       3.7
```

```
N1 =
         777.095382931401
N2 =
         1882.02854581854
N3 =
         3243.25753305588

N1rpm =
         7420.71428684531
N2rpm =
         17972.0487664244
N3rpm =
         30970.8280863522

Ns1c =
        0.656504606181855
Ns2c =
        0.643301879083287
Ns3c =
        0.651131920874496
Ns3E =
         0.59462735707489
Ns2E =
        0.603160188747609
Ns1E =
        0.589024644520001

Ds1c =
          3.8206024980233
Ds2c =
         3.85960921016809
Ds3c =
         3.83633258639107
Ds3E =
         3.05233181488388
Ds2E =
         3.00915085885995
Ds1E =
         3.08136512943198

D1c =
         1.15026303822927
D2c =
        0.470147150447466
D3c =
        0.274476812965902
D3E =
        0.279811267144404
D2E =
        0.523795099583716
D1E =
         1.26856670523217

vtip1c =
```

```
          446.932048082304
vtip2c =
          442.415178938687
vtip3c =
          445.099495650416
vtip3E =
                    453.75
vtip2E =
           492.89866478821
vtip1E =
           492.89866478821

Re1c =
          960955.860110237
Re2c =
          1348478.85329947
Re3c =
          1752935.67513927
Re3E =
          472634.279612791
Re2E =
           279070.10683235
Re1E =
          141415.886240055

f_eta_1c =
         0.968097378349482
f_eta_2c =
         0.975193765611316
f_eta_3c =
         0.979750639189225
f_eta_3E =
         0.934197773904176
f_eta_2E =
         0.902514063285964
f_eta_1E =
         0.834614585896914
```

Results Extracted From MATLAB Output:

Summary of Engine Model #2 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**178 horsepower**					
Output Power (SI units)	**132.8 kW**					
Thermal Input Power (from Nuclear Reactor)	**495.8 kW**					
Thermal Conversion Cycle Energy Efficiency	**0.2648**					
Air Mass Flow Rate (kg/sec)	**0.6684 kg/sec**					
	Low Pressure Stages		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**777.1 radians/sec**		**1,882.0 radians/sec**		**3,243.3 radians/sec**	
Shaft Speed (rpm)	**7,420.7 rpm**		**17,972.0 rpm**		**30,970.8 rpm**	
	Compressor	**Turbine**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**1.150 m**	**1.269 m**	**0.470 m**	**0.524**	**0.274 m**	**0.280 m**
Rotor Tip Speed (m/s)	**446.9 m/s**	**492.9 m/s**	**442.4 m/s**	**492.9 m/s**	**445.1 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.08**	**3.86**	**3.01**	**3.84**	**3.05**
Ns: Specific Speed	**0.657**	**0.589**	**0.643**	**0.603**	**0.651**	**0.595**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$9.61*10^5$**	**$1.41*10^5$**	**$1.35*10^6$**	**$2.79*10^5$**	**$1.75*10^6$**	**$4.73*10^5$**
Reynolds Number Efficiency Factor	**0.968**	**0.835**	**0.975**	**0.903**	**0.980**	**0.934**
Assumed Efficiency (from Balje*Reynolds)	**0.823**	**0.734**	**0.829**	**0.794**	**0.833**	**0.822**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**934.9 Kelvins**	**397.2 Kelvins**	**1126.7 Kelvins**	**555.6 Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,255 Pa**	**9,592 Pa**	**34,245 Pa**	**553.2Kelvins**	**102,974 Pa**
Pressure Ratio (Calculated)	**11.285**	**6.981**	**4.037**	**4.720**	**38,727 Pa**	**3.007**
Discharge Air Temperature (Kelvins)	**397.2 Kelvins**	**749.1 Kelvins**	**553.2Kelvins**	**934.9 Kelvins**	**695.7 Kelvins**	**1126.7 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**1,039 Pa**	**38,727 Pa**	**7,255 Pa**	**114,415 Pa**	**34,245 Pa**
Q:Volumetric Flow Rate On Low Pressure Side	**32.30 m^3/s**	**92.06 m^3/s**	**5.29 m^3/s**	**16.46 m^3/s**	**1.82 m^3/s**	**4.20 m^3/s**
Shaft Powers (kW) [Total=132.8 kW=178 hp]	**-103.1596 kW**	**+144.8207 kW**	**-102.40878 kW**	**+156.6025 kW**	**-101.9323**	**+137.3734 kW**

First we initialize variables in preparation for an iterative set of calculations.

```
clear
f_eta_1c=1;
f_eta_2c=1;
f_eta_3c=1;
f_eta_3E=1;
f_eta_2E=1;
f_eta_1E=1;
EngineModel=3
hp=[100 176 316 562 1000 1780 3160 5620 10000];
ep=0.9

EngineModel =
     3
ep =
                        0.9
```

The following MATLAB cell needs to be executed several times (e.g., perhaps 5) until its calculated quantities stop changing. Only then are its set of calculated quantities self-consistent. As implemented here, the user must manually initiate each of those iterative executions and decide when to stop iterating. .

```
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);
R=GasConstant('MarsMix');
T0=215;T4=1290;p0=850;
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);

Ec1=1.27e5
Ec2=1.27e5
Ec3=1.27e5
EE3=2.5e5
EE2=2.95e5
EE1=2.95e5
display(' ')
etac1=0.85*f_eta_1c
etac2=0.85*f_eta_2c
etac3=0.85*f_eta_3c
etaE3=0.88*f_eta_3E
etaE2=0.88*f_eta_2E
etaE1=0.88*f_eta_1E
display(' ')
T1c_out_s=interp1(h,T,Ec1+h0);
T1c_out=interp1(h,T,Ec1/etac1+h0);
pic1=exp((interp1(T,S0,T1c_out_s)-S00)/R)
p1c_out=p0*pic1;
p2c_in=p1c_out;
T2c_in=T1c_out;
T2c_out_s=interp1(h,T,Ec2+Ec1/etac1+h0);
```

```
T2c_out=interp1(h,T,Ec2/etac2+Ec1/etac1+h0);
pic2=exp((interp1(T,S0,T2c_out_s)-interp1(T,S0,T2c_in))/R)
p2c_out=p2c_in*pic2;
p3c_in=p2c_out;
T3c_in=T2c_out;
T3c_out_s=interp1(h,T,Ec3+Ec2/etac2+Ec1/etac1+h0);
T3c_out=interp1(h,T,Ec3/etac3+Ec2/etac2+Ec1/etac1+h0);
pic3=exp((interp1(T,S0,T3c_out_s)-interp1(T,S0,T3c_in))/R)
p3c_out=p3c_in*pic3;
p3E_in=ep*p3c_out;
T3E_in=T4;
T3E_out_s=interp1(h,T,h4-EE3);
T3E_out=interp1(h,T,h4-etaE3*EE3);
piE3=exp((S04-interp1(T,S0,T3E_out_s))/R)
p3E_out=p3E_in./piE3;
p2E_in=p3E_out;
T2E_in=T3E_out;
T2E_out_s=interp1(h,T,h4-etaE3*EE3-EE2);
T2E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2);
piE2=exp((interp1(T,S0,T3E_out)-interp1(T,S0,T2E_out_s))/R)
p2E_out=p2E_in/piE2;
p1E_in=p2E_out;
T1E_in=T2E_out;
T1E_out_s=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-EE1);
T1E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-etaE1*EE1);
piE1=exp((interp1(T,S0,T2E_out)-interp1(T,S0,T1E_out_s))/R)
p1E_out=p1E_in/piE1;
display(' ')
Wsp=EE1*etaE1+EE2*etaE2+EE3*etaE3-Ec1/etac1-Ec2/etac2-Ec3/etac3
HeatAdded=h4-(Ec3+Ec2/etac2+Ec1/etac1+h0)
CycleEfficiency=Wsp/HeatAdded
mdot=746*hp(EngineModel)/Wsp
ReactorPower=HeatAdded*mdot
display(' ')
ShaftPower1c=-Ec1/etac1*mdot
ShaftPower2c=-Ec2/etac2*mdot
ShaftPower3c=-Ec3/etac3*mdot
ShaftPower3E=+EE3*etaE3*mdot
ShaftPower2E=+EE2*etaE2*mdot
ShaftPower1E=+EE1*etaE1*mdot
display(' ')
Q1c=mdot*R*T0/p0
Q2c=mdot*R*T2c_in/p2c_in
Q3c=mdot*R*T3c_in/p3c_in
Q3E=mdot*R*T3E_out./p3E_out
Q2E=mdot*R*T2E_out/p2E_out
Q1E=mdot*R*T1E_out/p1E_out
display(' ')
display('                    Pressures   Temperatures')
display([p0 T0; p2c_in T2c_in; p3c_in T3c_in; p3c_out T3c_out; p3E_in
T3E_in; p2E_in T2E_in; p1E_in T1E_in; p1E_out T1E_out])
display(' ')
NsE_ideal=0.55, DsE_ideal=3.3
Nsc_ideal=0.70,Dsc_ideal=3.7
D1c_ideal=Dsc_ideal*Q1c^0.50/Ec1^0.25;
```

```
D2c_ideal=Dsc_ideal*Q2c^0.50/Ec2^0.25;
D3c_ideal=Dsc_ideal*Q3c^0.50/Ec3^0.25;
D3E_ideal=DsE_ideal*Q3E^0.50/EE3^0.25;
D2E_ideal=DsE_ideal*Q2E^0.50/EE2^0.25;
D1E_ideal=DsE_ideal*Q1E^0.50/EE1^0.25;
display(' ')
N1c_ideal=Nsc_ideal*Q1c^-0.50*Ec1^0.75;
N2c_ideal=Nsc_ideal*Q2c^-0.50*Ec2^0.75;
N3c_ideal=Nsc_ideal*Q3c^-0.50*Ec3^0.75;
N3E_ideal=NsE_ideal*Q3E^-0.50*EE3^0.75;
N2E_ideal=NsE_ideal*Q2E^-0.50*EE2^0.75;
N1E_ideal=NsE_ideal*Q1E^-0.50*EE1^0.75;
display(' ')
N1=(N1c_ideal+N1E_ideal)/2
N2=(N2c_ideal+N2E_ideal)/2
N3=(N3c_ideal+N3E_ideal)/2
display(' ')
N1rpm=N1*30/pi
N2rpm=N2*30/pi
N3rpm=N3*30/pi
display(' ')
Ns1c=N1*Q1c^0.5*Ec1^-0.75
Ns2c=N2*Q2c^0.5*Ec2^-0.75
Ns3c=N3*Q3c^0.5*Ec3^-0.75
Ns3E=N3*Q3E^0.5*EE3^-0.75
Ns2E=N2*Q2E^0.5*EE2^-0.75
Ns1E=N1*Q1E^0.5*EE1^-0.75
display(' ')
Ds1c= 10^(log10(Dsc_ideal)-0.5*log10(Ns1c/Nsc_ideal))
Ds2c= 10^(log10(Dsc_ideal)-0.5*log10(Ns2c/Nsc_ideal))
Ds3c= 10^(log10(Dsc_ideal)-0.5*log10(Ns3c/Nsc_ideal))
Ds3E= 10^(log10(DsE_ideal)-1.0*log10(Ns3E/NsE_ideal))
Ds2E= 10^(log10(DsE_ideal)-1.0*log10(Ns2E/NsE_ideal))
Ds1E= 10^(log10(DsE_ideal)-1.0*log10(Ns1E/NsE_ideal))
display(' ')
D1c=Ds1c*Q1c^0.50/Ec1^0.25
D2c=Ds2c*Q2c^0.50/Ec2^0.25
D3c=Ds3c*Q3c^0.50/Ec3^0.25
D3E=Ds3E*Q3E^0.50/EE3^0.25
D2E=Ds2E*Q2E^0.50/EE2^0.25
D1E=Ds1E*Q1E^0.50/EE1^0.25
display(' ')
vtip1c=N1*D1c/2
vtip2c=N2*D2c/2
vtip3c=N3*D3c/2
vtip3E=N3*D3E/2
vtip2E=N2*D2E/2
vtip1E=N1*D1E/2
display(' ')
Re1c=vtip1c*D1c*p0/R/T0/Viscosity('MarsMix',T0)
Re2c=vtip2c*D2c*p2c_in/R/T2c_in/ Viscosity('MarsMix',T2c_in)
Re3c=vtip3c*D3c*p3c_in/R/T3c_in/ Viscosity('MarsMix',T3c_in)
Re3E=vtip3E*D3E*p3E_out/R/T3E_out/ Viscosity('MarsMix',T3E_out)
Re2E=vtip2E*D2E*p2E_out/R/T2E_out/ Viscosity('MarsMix',T2E_out)
Re1E=vtip1E*D1E*p1E_out/R/T1E_out/ Viscosity('MarsMix',T1E_out)
```

```
log10Re_val=[3 3.5 4 4.5 5 5.5 6 6.5 7];
f_eta_val=[0.2 0.34 0.5 0.65 0.8 0.915 0.97 0.99 1.];
display(' ')
f_eta_1c=interp1(log10Re_val,f_eta_val,log10(Re1c),'linear','extrap' )
f_eta_2c=interp1(log10Re_val,f_eta_val,log10(Re2c),'linear','extrap' )
f_eta_3c=interp1(log10Re_val,f_eta_val,log10(Re3c),'linear','extrap' )
f_eta_3E=interp1(log10Re_val,f_eta_val,log10(Re3E),'linear','extrap' )
f_eta_2E=interp1(log10Re_val,f_eta_val,log10(Re2E),'linear','extrap' )
f_eta_1E=interp1(log10Re_val,f_eta_val,log10(Re1E),'linear','extrap' )

Ec1 =
      127000
Ec2 =
      127000
Ec3 =
      127000
EE3 =
      250000
EE2 =
      295000
EE1 =
      295000

etac1 =
         0.827626317506416
etac2 =
         0.832677049249739
etac3 =
         0.836576157064987
etaE3 =
          0.83289208183147
etaE2 =
         0.810969867904008
etaE1 =
         0.757868696967379

pic1 =
          11.2851768923859
pic2 =
          4.04891735845423
pic3 =
          2.96132341685378
piE3 =
          3.00698371395171
piE2 =
          4.73708169056052
piE1 =
          7.09592708264188

Wsp =
          213250.134005911
HeatAdded =
          743318.466309685
CycleEfficiency =
         0.286889326273062
```

```
mdot =
          1.10544361952647
ReactorPower =
          821696.655858245

ShaftPower1c =
         -169631.313927827
ShaftPower2c =
         -168602.388892978
ShaftPower3c =
         -167816.568156097
ShaftPower3E =
           230178.80940368
ShaftPower2E =
            264462.0325003
ShaftPower1E =
          247145.429072922

Q1c =
          53.4228122312782
Q2c =
          8.72431865620647
Q3c =
           3.0000152924865
Q3E =
             6.89959850526
Q2E =
          26.9860342506419
Q1E =
          151.832868279728

                  Pressures   Temperatures
ans =
                       850                       215
          9592.40035852798          396.233877730709
          38838.8363208865          551.673935848888
          115014.355480392          693.642498032305
          103512.919932353                      1290
          34424.1704576175          1124.55342135996
           7266.9573180918          928.505466266397
          1024.10259201624          736.210572507423

NsE_ideal =
                      0.55
DsE_ideal =
                       3.3
Nsc_ideal =
                       0.7
Dsc_ideal =
                       3.7


N1 =
          604.648105359703
N2 =
```

```
          1467.26223271581
N3 =
           2529.9488312716

N1rpm =
          5773.96408794875
N2rpm =
          14011.3222289263
N3rpm =
          24159.2317359863

Ns1c =
          0.65692120167349
Ns2c =
         0.644199981711374
Ns3c =
         0.651359394199942
Ns3E =
         0.594386094651297
Ns2E =
         0.602158415986668
Ns1E =
         0.588598407262829

Ds1c =
          3.81939086244742
Ds2c =
          3.85691786101497
Ds3c =
          3.83566264937798
Ds3E =
          3.05357076205625
Ds2E =
          3.01415699226927
Ds1E =
          3.08359651946788

D1c =
          1.47879014444276
D2c =
         0.603469353257317
D3c =
         0.351925881856552
D3E =
         0.358702906866253
D2E =
         0.671861721508207
D1E =
          1.63036536596765

vtip1c =
          447.073829530958
vtip2c =
          442.723895317949
vtip3c =
```

```
          445.177236748604
vtip3E =
                   453.75
vtip2E =
           492.89866478821
vtip1E =
           492.89866478821

Re1c =
          1235806.77816702
Re2c =
          1739808.70929441
Re3c =
          2265589.05824278
Re3E =
            611047.9246931
Re2E =
          362748.910061588
Re1E =
          184565.160644745

f_eta_1c =
         0.973678022916336
f_eta_2c =
         0.979620060021768
f_eta_3c =
         0.984207245542854
f_eta_3E =
         0.946468280078382
f_eta_2E =
         0.921556672782325
f_eta_1E =
         0.861214436757786
```

Results Extracted From MATLAB Output:

Summary of Engine Model #3 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**316 horsepower**					
Output Power (SI units)	**235.7 kW**					
Thermal Input Power (from Nuclear Reactor)	**821.7 kW**					
Thermal Conversion Cycle Energy Efficiency	**0.2869**					
Air Mass Flow Rate (kg/sec)	**1.1054 kg/sec**					
	Low Pressure Stages		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**604.6 radians/sec**		**1,467.3 radians/sec**		**2,529.9 radians/sec**	
Shaft Speed (rpm)	**5774.0 rpm**		**14,011.3 rpm**		**24,159.2 rpm**	
	Compressor	**Turbine**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**1.479 m**	**1.630 m**	**0.603 m**	**0.672**	**0.352 m**	**0.359 m**
Rotor Tip Speed (m/s)	**447.1 m/s**	**492.9 m/s**	**442.7 m/s**	**492.9 m/s**	**445.2 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.08**	**3.86**	**3.01**	**3.84**	**3.05**
Ns: Specific Speed	**0.657**	**0.589**	**0.644**	**0.602**	**0.651**	**0.594**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$1.24*10^6$**	**$2.27*10^5$**	**$1.74*10^6$**	**$3.63*10^5$**	**$2.27*10^6$**	**$6.11*10^5$**
Reynolds Number Efficiency Factor	**0.974**	**0.861**	**0.980**	**0.922**	**0.984**	**0.946**
Assumed Efficiency (from Balje*Reynolds)	**0.828**	**0.758**	**0.833**	**0.811**	**0.837**	**0.833**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**928.5 Kelvins**	**396.2 Kelvins**	**1124.6 Kelvins**	**551.7Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,267 Pa**	**9,592 Pa**	**34,424 Pa**	**38,839 Pa**	**103,513Pa**
Pressure Ratio (Calculated)	**11.285**	**7.096**	**4.049**	**4.737**	**2.961**	**3.007**
Discharge Air Temperature (Kelvins)	**396.2 Kelvins**	**736.2 Kelvins**	**551.7Kelvins**	**928.5 Kelvins**	**693.6 Kelvins**	**1124.6 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**1,024 Pa**	**38,839 Pa**	**7,267 Pa**	**115,014 Pa**	**34,424 Pa**
Q:Volumetric Flow Rate On Low Pressure Side	**53.42 m^3/s**	**151.83 m^3/s**	**8.72 m^3/s**	**26.99 m^3/s**	**3.00 m^3/s**	**6.90 m^3/s**
Shaft Powers (kW) [Total=235.7 kW=316 hp]	**-169.6313 kW**	**+247.1454 kW**	**-168.6024 kW**	**+264.4620 kW**	**-167.8166**	**+230.1788 kW**

First we initialize variables in preparation for an iterative set of calculations.

```
clear
f_eta_1c=1;
f_eta_2c=1;
f_eta_3c=1;
f_eta_3E=1;
f_eta_2E=1;
f_eta_1E=1;
EngineModel=4
hp=[100 176 316 562 1000 1780 3160 5620 10000];
ep=0.9

EngineModel =
     4
ep =
                        0.9
```

The following MATLAB cell needs to be executed several times (e.g., perhaps 5) until its calculated quantities stop changing. Only then are its set of calculated quantities self-consistent. As implemented here, the user must manually initiate each of those iterative executions and decide when to stop iterating. .

```
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);
R=GasConstant('MarsMix');
T0=215;T4=1290;p0=850;
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);

Ec1=1.27e5
Ec2=1.27e5
Ec3=1.27e5
EE3=2.5e5
EE2=2.95e5
EE1=2.95e5
display(' ')
etac1=0.85*f_eta_1c
etac2=0.85*f_eta_2c
etac3=0.85*f_eta_3c
etaE3=0.88*f_eta_3E
etaE2=0.88*f_eta_2E
etaE1=0.88*f_eta_1E
display(' ')
T1c_out_s=interp1(h,T,Ec1+h0);
T1c_out=interp1(h,T,Ec1/etac1+h0);
pic1=exp((interp1(T,S0,T1c_out_s)-S00)/R)
p1c_out=p0*pic1;
p2c_in=p1c_out;
T2c_in=T1c_out;
T2c_out_s=interp1(h,T,Ec2+Ec1/etac1+h0);
T2c_out=interp1(h,T,Ec2/etac2+Ec1/etac1+h0);
```

```
pic2=exp((interp1(T,S0,T2c_out_s)-interp1(T,S0,T2c_in))/R)
p2c_out=p2c_in*pic2;
p3c_in=p2c_out;
T3c_in=T2c_out;
T3c_out_s=interp1(h,T,Ec3+Ec2/etac2+Ec1/etac1+h0);
T3c_out=interp1(h,T,Ec3/etac3+Ec2/etac2+Ec1/etac1+h0);
pic3=exp((interp1(T,S0,T3c_out_s)-interp1(T,S0,T3c_in))/R)
p3c_out=p3c_in*pic3;
p3E_in=ep*p3c_out;
T3E_in=T4;
T3E_out_s=interp1(h,T,h4-EE3);
T3E_out=interp1(h,T,h4-etaE3*EE3);
piE3=exp((S04-interp1(T,S0,T3E_out_s))/R)
p3E_out=p3E_in./piE3;
p2E_in=p3E_out;
T2E_in=T3E_out;
T2E_out_s=interp1(h,T,h4-etaE3*EE3-EE2);
T2E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2);
piE2=exp((interp1(T,S0,T3E_out)-interp1(T,S0,T2E_out_s))/R)
p2E_out=p2E_in/piE2;
p1E_in=p2E_out;
T1E_in=T2E_out;
T1E_out_s=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-EE1);
T1E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-etaE1*EE1);
piE1=exp((interp1(T,S0,T2E_out)-interp1(T,S0,T1E_out_s))/R)
p1E_out=p1E_in/piE1;
display(' ')
Wsp=EE1*etaE1+EE2*etaE2+EE3*etaE3-Ec1/etac1-Ec2/etac2-Ec3/etac3
HeatAdded=h4-(Ec3+Ec2/etac2+Ec1/etac1+h0)
CycleEfficiency=Wsp/HeatAdded
mdot=746*hp(EngineModel)/Wsp
ReactorPower=HeatAdded*mdot
display(' ')
ShaftPower1c=-Ec1/etac1*mdot
ShaftPower2c=-Ec2/etac2*mdot
ShaftPower3c=-Ec3/etac3*mdot
ShaftPower3E=+EE3*etaE3*mdot
ShaftPower2E=+EE2*etaE2*mdot
ShaftPower1E=+EE1*etaE1*mdot
display(' ')
Q1c=mdot*R*T0/p0
Q2c=mdot*R*T2c_in/p2c_in
Q3c=mdot*R*T3c_in/p3c_in
Q3E=mdot*R*T3E_out./p3E_out
Q2E=mdot*R*T2E_out/p2E_out
Q1E=mdot*R*T1E_out/p1E_out
display(' ')
display('                    Pressures   Temperatures')
display([p0 T0; p2c_in T2c_in; p3c_in T3c_in; p3c_out T3c_out; p3E_in
T3E_in; p2E_in T2E_in; p1E_in T1E_in; p1E_out T1E_out])
display(' ')
NsE_ideal=0.55, DsE_ideal=3.3
Nsc_ideal=0.70,Dsc_ideal=3.7
D1c_ideal=Dsc_ideal*Q1c^0.50/Ec1^0.25;
D2c_ideal=Dsc_ideal*Q2c^0.50/Ec2^0.25;
```

```
D3c_ideal=Dsc_ideal*Q3c^0.50/Ec3^0.25;
D3E_ideal=DsE_ideal*Q3E^0.50/EE3^0.25;
D2E_ideal=DsE_ideal*Q2E^0.50/EE2^0.25;
D1E_ideal=DsE_ideal*Q1E^0.50/EE1^0.25;
display(' ')
N1c_ideal=Nsc_ideal*Q1c^-0.50*Ec1^0.75;
N2c_ideal=Nsc_ideal*Q2c^-0.50*Ec2^0.75;
N3c_ideal=Nsc_ideal*Q3c^-0.50*Ec3^0.75;
N3E_ideal=NsE_ideal*Q3E^-0.50*EE3^0.75;
N2E_ideal=NsE_ideal*Q2E^-0.50*EE2^0.75;
N1E_ideal=NsE_ideal*Q1E^-0.50*EE1^0.75;
display(' ')
N1=(N1c_ideal+N1E_ideal)/2
N2=(N2c_ideal+N2E_ideal)/2
N3=(N3c_ideal+N3E_ideal)/2
display(' ')
N1rpm=N1*30/pi
N2rpm=N2*30/pi
N3rpm=N3*30/pi
display(' ')
Ns1c=N1*Q1c^0.5*Ec1^-0.75
Ns2c=N2*Q2c^0.5*Ec2^-0.75
Ns3c=N3*Q3c^0.5*Ec3^-0.75
Ns3E=N3*Q3E^0.5*EE3^-0.75
Ns2E=N2*Q2E^0.5*EE2^-0.75
Ns1E=N1*Q1E^0.5*EE1^-0.75
display(' ')
Ds1c= 10^(log10(Dsc_ideal)-0.5*log10(Ns1c/Nsc_ideal))
Ds2c= 10^(log10(Dsc_ideal)-0.5*log10(Ns2c/Nsc_ideal))
Ds3c= 10^(log10(Dsc_ideal)-0.5*log10(Ns3c/Nsc_ideal))
Ds3E= 10^(log10(DsE_ideal)-1.0*log10(Ns3E/NsE_ideal))
Ds2E= 10^(log10(DsE_ideal)-1.0*log10(Ns2E/NsE_ideal))
Ds1E= 10^(log10(DsE_ideal)-1.0*log10(Ns1E/NsE_ideal))
display(' ')
D1c=Ds1c*Q1c^0.50/Ec1^0.25
D2c=Ds2c*Q2c^0.50/Ec2^0.25
D3c=Ds3c*Q3c^0.50/Ec3^0.25
D3E=Ds3E*Q3E^0.50/EE3^0.25
D2E=Ds2E*Q2E^0.50/EE2^0.25
D1E=Ds1E*Q1E^0.50/EE1^0.25
display(' ')
vtip1c=N1*D1c/2
vtip2c=N2*D2c/2
vtip3c=N3*D3c/2
vtip3E=N3*D3E/2
vtip2E=N2*D2E/2
vtip1E=N1*D1E/2
display(' ')
Re1c=vtip1c*D1c*p0/R/T0/Viscosity('MarsMix',T0)
Re2c=vtip2c*D2c*p2c_in/R/T2c_in/ Viscosity('MarsMix',T2c_in)
Re3c=vtip3c*D3c*p3c_in/R/T3c_in/ Viscosity('MarsMix',T3c_in)
Re3E=vtip3E*D3E*p3E_out/R/T3E_out/ Viscosity('MarsMix',T3E_out)
Re2E=vtip2E*D2E*p2E_out/R/T2E_out/ Viscosity('MarsMix',T2E_out)
Re1E=vtip1E*D1E*p1E_out/R/T1E_out/ Viscosity('MarsMix',T1E_out)
log10Re_val=[3 3.5 4 4.5 5 5.5 6 6.5 7];
```

```
f_eta_val=[0.2 0.34 0.5 0.65 0.8 0.915 0.97 0.99 1.];
display(' ')
f_eta_1c=interp1(log10Re_val,f_eta_val,log10(Re1c),'linear','extrap' )
f_eta_2c=interp1(log10Re_val,f_eta_val,log10(Re2c),'linear','extrap' )
f_eta_3c=interp1(log10Re_val,f_eta_val,log10(Re3c),'linear','extrap' )
f_eta_3E=interp1(log10Re_val,f_eta_val,log10(Re3E),'linear','extrap' )
f_eta_2E=interp1(log10Re_val,f_eta_val,log10(Re2E),'linear','extrap' )
f_eta_1E=interp1(log10Re_val,f_eta_val,log10(Re1E),'linear','extrap' )

Ec1 =
      127000
Ec2 =
      127000
Ec3 =
      127000
EE3 =
      250000
EE2 =
      295000
EE1 =
      295000

etac1 =
         0.831375526882763
etac2 =
         0.836463866678043
etac3 =
         0.840388259450675
etaE3 =
         0.843772132702296
etaE2 =
          0.82197904807455
etaE1 =
         0.781440715586123

pic1 =
          11.2851768923859
pic2 =
           4.0580983159561
pic3 =
          2.96744586404023
piE3 =
          3.00698371395171
piE2 =
           4.7541806749214
piE1 =
          7.18830836016151

Wsp =
          228242.719743775
HeatAdded =
          744700.960635104
CycleEfficiency =
          0.30648908999543
mdot =
```

```
          1.83686910351687
ReactorPower =
          1367918.18594995

ShaftPower1c =
         -280598.079451933
ShaftPower2c =
         -278891.157693525
ShaftPower3c =
         -277588.809128686
ShaftPower3E =
          387474.740242345
ShaftPower2E =
          445411.035558172
ShaftPower1E =
          423444.270473627

Q1c =
          88.7704370238738
Q2c =
          14.4694936294625
Q3c =
          4.96162174394561
Q3E =
          11.3930067984063
Q2E =
           44.567928540935
Q1E =
          251.399520233173

                  Pressures   Temperatures
ans =
                       850                        215
          9592.40035852798          395.486629723149
          38926.9037409191          550.331902716375
          115513.479505883          691.768310427843
          103962.131555294                      1290
          34573.5599008848          1122.36375192101
          7272.24358200404          923.510935551275
          1011.67663066706          724.698460740249

NsE_ideal =
                     0.55
DsE_ideal =
                      3.3
Nsc_ideal =
                      0.7
Dsc_ideal =
                      3.7


N1 =
          469.453418894837
N2 =
          1140.42537181434
```

```
N3 =
         1967.97823392231

N1rpm =
         4482.94993011021
N2rpm =
         10890.2601090999
N3rpm =
         18792.8078295596

Ns1c =
        0.657466901875513
Ns2c =
        0.644823336658041
Ns3c =
        0.651597375116975
Ns3E =
        0.594134077220232
Ns2E =
        0.601466693888748
Ns1E =
        0.588041824706614

Ds1c =
          3.817805478437
Ds2c =
         3.85505315636038
Ds3c =
         3.83496214174019
Ds3E =
          3.0548660135635
Ds2E =
         3.01762345021172
Ds1E =
         3.08651514865552

D1c =
         1.90544775543061
D2c =
         0.77679443756136
D3c =
         0.45250353556697
D3E =
        0.461133148912575
D2E =
        0.864411958853639
D1E =
         2.09988316177808

vtip1c =
         447.259481656196
vtip2c =
         442.938042639611
vtip3c =
         445.258554384343
```

```
vtip3E =
                  453.75
vtip2E =
           492.89866478821
vtip1E =
           492.89866478821

Re1c =
          1593020.55235849
Re2c =
          2248425.13549008
Re3c =
          2932930.34241075
Re3E =
          791539.232132992
Re2E =
          471343.229874981
Re1E =
          241330.510258332

f_eta_1c =
         0.978088855155543
f_eta_2c =
         0.984075137267711
f_eta_3c =
          0.98869206994144
f_eta_3E =
         0.958831968978439
f_eta_2E =
         0.934067100083444
f_eta_1E =
         0.888000813163833
```

Results Extracted From MATLAB Output:

Summary of Engine Model #4 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**562 horsepower**					
Output Power (SI units)	**419.3 kW**					
Thermal Input Power (from Nuclear Reactor)	**1.368 MW**					
Thermal Conversion Cycle Energy Efficiency	**0.3065**					
Air Mass Flow Rate (kg/sec)	**1.837 kg/sec**					
	Low Pressure Stages		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**469.5 radians/sec**		**1,140.4 radians/sec**		**1,968.0 radians/sec**	
Shaft Speed (rpm)	**4,482,9 rpm**		**10,890.3 rpm**		**18,792.8 rpm**	
	Compressor	**Turbine**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**1.905 m**	**2.100 m**	**0.777 m**	**0.864**	**0.453 m**	**0.461 m**
Rotor Tip Speed (m/s)	**447.3 m/s**	**492.9 m/s**	**442.9 m/s**	**492.9 m/s**	**445.3 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.09**	**3.86**	**3.02**	**3.83**	**3.05**
Ns: Specific Speed	**0.657**	**0.588**	**0.645**	**0.601**	**0.652**	**0.594**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$1.59*10^6$**	**$2.41*10^5$**	**$2.25*10^6$**	**$4.71*10^5$**	**$2.93*10^6$**	**$7.92*10^5$**
Reynolds Number Efficiency Factor	**0.978**	**0.888**	**0.984**	**0.934**	**0.989**	**0.959**
Assumed Efficiency (from Balje*Reynolds)	**0.831**	**0.781**	**0.836**	**0.822**	**0.840**	**0.844**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**923.5 Kelvins**	**395.5 Kelvins**	**1122.3 Kelvins**	**550.3 Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,272 Pa**	**9,592 Pa**	**34,574 Pa**	**38,927 Pa**	**103,962 Pa**
Pressure Ratio (Calculated)	**11.285**	**7.188**	**4.058**	**4.754**	**2.967**	**3.007**
Discharge Air Temperature (Kelvins)	**395.5 Kelvins**	**724.7 Kelvins**	**550.3 Kelvins**	**923.5 Kelvins**	**691.8 Kelvins**	**1122.3 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**1,012 Pa**	**38,927 Pa**	**7,272 Pa**	**115,513 Pa**	**34,574 Pa**
Q:Volumetric Flow Rate On Low Pressure Side	**88.77 m^3/s**	**251.40 m^3/s**	**14.47 m^3/s**	**44.57 m^3/s**	**4.96 m^3/s**	**11.39 m^3/s**
Shaft Powers (kW) [Total=419.3 kW=562 hp]	**-280.598 kW**	**+423.444 kW**	**-278.891 kW**	**+445.411 kW**	**-277.589 kW**	**+387.475 kW**

First we initialize variables in preparation for an iterative set of calculations.

```
clear
f_eta_1c=1;
f_eta_2c=1;
f_eta_3c=1;
f_eta_3E=1;
f_eta_2E=1;
f_eta_1E=1;
EngineModel=5
hp=[100 176 316 562 1000 1780 3160 5620 10000];
ep=0.9
```

```
EngineModel =
     5
ep =
                        0.9
```

The following MATLAB cell needs to be executed several times (e.g., perhaps 5) until its calculated quantities stop changing. Only then are its set of calculated quantities self-consistent. As implemented here, the user must manually initiate each of those iterative executions and decide when to stop iterating. .

```
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);
R=GasConstant('MarsMix');
T0=215;T4=1290;p0=850;
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);

Ec1=1.27e5
Ec2=1.27e5
Ec3=1.27e5
EE3=2.5e5
EE2=2.95e5
EE1=2.95e5
display(' ')
etac1=0.85*f_eta_1c
etac2=0.85*f_eta_2c
etac3=0.85*f_eta_3c
etaE3=0.88*f_eta_3E
etaE2=0.88*f_eta_2E
etaE1=0.88*f_eta_1E
display(' ')
T1c_out_s=interp1(h,T,Ec1+h0);
T1c_out=interp1(h,T,Ec1/etac1+h0);
pic1=exp((interp1(T,S0,T1c_out_s)-S00)/R)
p1c_out=p0*pic1;
p2c_in=p1c_out;
T2c_in=T1c_out;
T2c_out_s=interp1(h,T,Ec2+Ec1/etac1+h0);
T2c_out=interp1(h,T,Ec2/etac2+Ec1/etac1+h0);
```

```
pic2=exp((interp1(T,S0,T2c_out_s)-interp1(T,S0,T2c_in))/R)
p2c_out=p2c_in*pic2;
p3c_in=p2c_out;
T3c_in=T2c_out;
T3c_out_s=interp1(h,T,Ec3+Ec2/etac2+Ec1/etac1+h0);
T3c_out=interp1(h,T,Ec3/etac3+Ec2/etac2+Ec1/etac1+h0);
pic3=exp((interp1(T,S0,T3c_out_s)-interp1(T,S0,T3c_in))/R)
p3c_out=p3c_in*pic3;
p3E_in=ep*p3c_out;
T3E_in=T4;
T3E_out_s=interp1(h,T,h4-EE3);
T3E_out=interp1(h,T,h4-etaE3*EE3);
piE3=exp((S04-interp1(T,S0,T3E_out_s))/R)
p3E_out=p3E_in./piE3;
p2E_in=p3E_out;
T2E_in=T3E_out;
T2E_out_s=interp1(h,T,h4-etaE3*EE3-EE2);
T2E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2);
piE2=exp((interp1(T,S0,T3E_out)-interp1(T,S0,T2E_out_s))/R)
p2E_out=p2E_in/piE2;
p1E_in=p2E_out;
T1E_in=T2E_out;
T1E_out_s=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-EE1);
T1E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-etaE1*EE1);
piE1=exp((interp1(T,S0,T2E_out)-interp1(T,S0,T1E_out_s))/R)
p1E_out=p1E_in/piE1;
display(' ')
Wsp=EE1*etaE1+EE2*etaE2+EE3*etaE3-Ec1/etac1-Ec2/etac2-Ec3/etac3
HeatAdded=h4-(Ec3+Ec2/etac2+Ec1/etac1+h0)
CycleEfficiency=Wsp/HeatAdded
mdot=746*hp(EngineModel)/Wsp
ReactorPower=HeatAdded*mdot
display(' ')
ShaftPower1c=-Ec1/etac1*mdot
ShaftPower2c=-Ec2/etac2*mdot
ShaftPower3c=-Ec3/etac3*mdot
ShaftPower3E=+EE3*etaE3*mdot
ShaftPower2E=+EE2*etaE2*mdot
ShaftPower1E=+EE1*etaE1*mdot
display(' ')
Q1c=mdot*R*T0/p0
Q2c=mdot*R*T2c_in/p2c_in
Q3c=mdot*R*T3c_in/p3c_in
Q3E=mdot*R*T3E_out./p3E_out
Q2E=mdot*R*T2E_out/p2E_out
Q1E=mdot*R*T1E_out/p1E_out
display(' ')
display('                    Pressures   Temperatures')
display([p0 T0; p2c_in T2c_in; p3c_in T3c_in; p3c_out T3c_out; p3E_in
T3E_in; p2E_in T2E_in; p1E_in T1E_in; p1E_out T1E_out])
display(' ')
NsE_ideal=0.55, DsE_ideal=3.3
Nsc_ideal=0.70,Dsc_ideal=3.7
D1c_ideal=Dsc_ideal*Q1c^0.50/Ec1^0.25;
D2c_ideal=Dsc_ideal*Q2c^0.50/Ec2^0.25;
```

```
D3c_ideal=Dsc_ideal*Q3c^0.50/Ec3^0.25;
D3E_ideal=DsE_ideal*Q3E^0.50/EE3^0.25;
D2E_ideal=DsE_ideal*Q2E^0.50/EE2^0.25;
D1E_ideal=DsE_ideal*Q1E^0.50/EE1^0.25;
display(' ')
N1c_ideal=Nsc_ideal*Q1c^-0.50*Ec1^0.75;
N2c_ideal=Nsc_ideal*Q2c^-0.50*Ec2^0.75;
N3c_ideal=Nsc_ideal*Q3c^-0.50*Ec3^0.75;
N3E_ideal=NsE_ideal*Q3E^-0.50*EE3^0.75;
N2E_ideal=NsE_ideal*Q2E^-0.50*EE2^0.75;
N1E_ideal=NsE_ideal*Q1E^-0.50*EE1^0.75;
display(' ')
N1=(N1c_ideal+N1E_ideal)/2
N2=(N2c_ideal+N2E_ideal)/2
N3=(N3c_ideal+N3E_ideal)/2
display(' ')
N1rpm=N1*30/pi
N2rpm=N2*30/pi
N3rpm=N3*30/pi
display(' ')
Ns1c=N1*Q1c^0.5*Ec1^-0.75
Ns2c=N2*Q2c^0.5*Ec2^-0.75
Ns3c=N3*Q3c^0.5*Ec3^-0.75
Ns3E=N3*Q3E^0.5*EE3^-0.75
Ns2E=N2*Q2E^0.5*EE2^-0.75
Ns1E=N1*Q1E^0.5*EE1^-0.75
display(' ')
Ds1c= 10^(log10(Dsc_ideal)-0.5*log10(Ns1c/Nsc_ideal))
Ds2c= 10^(log10(Dsc_ideal)-0.5*log10(Ns2c/Nsc_ideal))
Ds3c= 10^(log10(Dsc_ideal)-0.5*log10(Ns3c/Nsc_ideal))
Ds3E= 10^(log10(DsE_ideal)-1.0*log10(Ns3E/NsE_ideal))
Ds2E= 10^(log10(DsE_ideal)-1.0*log10(Ns2E/NsE_ideal))
Ds1E= 10^(log10(DsE_ideal)-1.0*log10(Ns1E/NsE_ideal))
display(' ')
D1c=Ds1c*Q1c^0.50/Ec1^0.25
D2c=Ds2c*Q2c^0.50/Ec2^0.25
D3c=Ds3c*Q3c^0.50/Ec3^0.25
D3E=Ds3E*Q3E^0.50/EE3^0.25
D2E=Ds2E*Q2E^0.50/EE2^0.25
D1E=Ds1E*Q1E^0.50/EE1^0.25
display(' ')
vtip1c=N1*D1c/2
vtip2c=N2*D2c/2
vtip3c=N3*D3c/2
vtip3E=N3*D3E/2
vtip2E=N2*D2E/2
vtip1E=N1*D1E/2
display(' ')
Re1c=vtip1c*D1c*p0/R/T0/Viscosity('MarsMix',T0)
Re2c=vtip2c*D2c*p2c_in/R/T2c_in/ Viscosity('MarsMix',T2c_in)
Re3c=vtip3c*D3c*p3c_in/R/T3c_in/ Viscosity('MarsMix',T3c_in)
Re3E=vtip3E*D3E*p3E_out/R/T3E_out/ Viscosity('MarsMix',T3E_out)
Re2E=vtip2E*D2E*p2E_out/R/T2E_out/ Viscosity('MarsMix',T2E_out)
Re1E=vtip1E*D1E*p1E_out/R/T1E_out/ Viscosity('MarsMix',T1E_out)
log10Re_val=[3 3.5 4 4.5 5 5.5 6 6.5 7];
```

```
f_eta_val=[0.2 0.34 0.5 0.65 0.8 0.915 0.97 0.99 1.];
display(' ')
f_eta_1c=interp1(log10Re_val,f_eta_val,log10(Re1c),'linear','extrap' )
f_eta_2c=interp1(log10Re_val,f_eta_val,log10(Re2c),'linear','extrap' )
f_eta_3c=interp1(log10Re_val,f_eta_val,log10(Re3c),'linear','extrap' )
f_eta_3E=interp1(log10Re_val,f_eta_val,log10(Re3E),'linear','extrap' )
f_eta_2E=interp1(log10Re_val,f_eta_val,log10(Re2E),'linear','extrap' )
f_eta_1E=interp1(log10Re_val,f_eta_val,log10(Re1E),'linear','extrap' )

Ec1 =
      127000
Ec2 =
      127000
Ec3 =
      127000
EE3 =
      250000
EE2 =
      295000
EE1 =
      295000

etac1 =
          0.83516892420486
etac2 =
         0.840295076493684
etac3 =
         0.842872471339531
etaE3 =
         0.854026147204705
etaE2 =
           0.8331165863324
etaE1 =
         0.805265196623003

pic1 =
          11.2851768923859
pic2 =
          4.06736056774146
pic3 =
          2.97361331174355
piE3 =
          3.00698371395171
piE2 =
          4.77042786268042
piE1 =
          7.28121946254572

Wsp =
          242951.507534219
HeatAdded =
          746087.048802538
CycleEfficiency =
         0.325634264693582
mdot =
```

```
          3.07057160324443
ReactorPower =
          2290913.70560152

ShaftPower1c =
         -466926.608869355
ShaftPower2c =
         -464078.160780434
ShaftPower3c =
         -462659.069873639
ShaftPower3E =
          655587.109008755
ShaftPower2E =
          754652.518994341
ShaftPower1E =
          729424.211520333

Q1c =
          148.391620617512
Q2c =
          24.1418396833416
Q3c =
          8.25487838989705
Q3E =
          18.9272721498817
Q2E =
            74.03588853942
Q1E =
          418.514009026336

                  Pressures   Temperatures
ans =
                       850                        215
          9592.40035852798          394.736980383954
          39015.7509682658           548.98541489351
          116017.756446906          690.109716709069
          104415.980802216                      1290
          34724.4916285212          1120.29933817828
          7279.11471006086          918.609669593713
          999.710934068711          713.172516201674

NsE_ideal =
                      0.55
DsE_ideal =
                       3.3
Nsc_ideal =
                       0.7
Dsc_ideal =
                       3.7


N1 =
          363.447809101185
N2 =
          883.776246822166
```

```
N3 =
          1526.24519328994

N1rpm =
          3470.67092246238
N2rpm =
          8439.44149613704
N3rpm =
          14574.5680129403

Ns1c =
         0.658103125613643
Ns2c =
         0.645467997937523
Ns3c =
         0.651818763861661
Ns3E =
         0.593899987424626
Ns2E =
         0.600754398689066
Ns1E =
         0.587395402702588

Ds1c =
          3.81595959310505
Ds2c =
          3.85312755792434
Ds3c =
          3.83431081839273
Ds3E =
          3.05607011017212
Ds2E =
          3.02120134943763
Ds1E =
          3.08991182370383

D1c =
          2.46239389672048
D2c =
           1.0028769211279
D3c =
         0.583568342926633
D3E =
         0.594596467192677
D2E =
          1.11543768360045
D1E =
          2.71234907706369

vtip1c =
          447.475833453594
vtip2c =
          443.159400689493
vtip3c =
          445.334189173973
```

```
vtip3E =
                  453.75
vtip2E =
           492.89866478821
vtip1E =
           492.89866478821

Re1c =
          2059642.70910504
Re2c =
          2914479.96173446
Re3c =
          3808321.53459323
Re3E =
          1028268.31819965
Re2E =
          614319.763177994
Re1E =
          316718.563563045

f_eta_1c =
         0.982551675554142
f_eta_2c =
           0.9885824429484
f_eta_3c =
         0.991614672170323
f_eta_3E =
         0.970484258201488
f_eta_2E =
         0.946723393587511
f_eta_1E =
         0.915074087096706
```

Results Extracted From MATLAB Output:

Summary of Engine Model #5 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**1000 horsepower**					
Output Power (SI units)	**746.0 kW**					
Thermal Input Power (from Nuclear Reactor)	**2.291 MW**					
Thermal Conversion Cycle Energy Efficiency	**0.3256**					
Air Mass Flow Rate (kg/sec)	**3.071 kg/sec**					
	Low Pressure Stages		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**363.4 radians/sec**		**883.8 radians/sec**		**1,526.2 radians/sec**	
Shaft Speed (rpm)	**3,470.7 rpm**		**8,439.4 rpm**		**13,574.6 rpm**	
	Compressor	**Turbine**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**2.462 m**	**2.712 m**	**1.003 m**	**1.115**	**0.584 m**	**0.595 m**
Rotor Tip Speed (m/s)	**447.5 m/s**	**492.9 m/s**	**443.2 m/s**	**492.9 m/s**	**445.3 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.09**	**3.85**	**3.02**	**3.83**	**3.06**
Ns: Specific Speed	**0.658**	**0.587**	**0.645**	**0.600**	**0.652**	**0.594**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$2.06*10^6$**	**$3.17*10^5$**	**$2.91*10^6$**	**$6.14*10^5$**	**$3.81*10^6$**	**$1.03*10^6$**
Reynolds Number Efficiency Factor	**0.983**	**0.915**	**0.989**	**0.946**	**0.992**	**0.970**
Assumed Efficiency (from Balje*Reynolds)	**0.835**	**0.805**	**0.840**	**0.833**	**0.843**	**0.854**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**918.6 Kelvins**	**394.7 Kelvins**	**1120.3 Kelvins**	**549.0Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,279 Pa**	**9,592 Pa**	**34,724 Pa**	**39,016 Pa**	**104,416 Pa**
Pressure Ratio (Calculated)	**11.285**	**7.281**	**4.067**	**4.770**	**2.974**	**3.007**
Discharge Air Temperature (Kelvins)	**394.7 Kelvins**	**713.2 Kelvins**	**549.0Kelvins**	**918.6 Kelvins**	**690.1 Kelvins**	**1120.3 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**1,000 Pa**	**39,016 Pa**	**7,279 Pa**	**116,018 Pa**	**34,724 Pa**
Q:Volumetric Flow Rate On Low Pressure Side	**148.39 m^3/s**	**418.51 m^3/s**	**24.14 m^3/s**	**74.94 m^3/s**	**8.25 m^3/s**	**18.93 m^3/s**
Shaft Powers (kW) [Total=746 kW=1000 hp]	**-466.927 kW**	**+729.424 kW**	**-464.078 kW**	**+754.653 kW**	**-462.659 kW**	**+655.587 kW**

First we initialize variables in preparation for an iterative set of calculations.

```
clear
f_eta_1c=1;
f_eta_2c=1;
f_eta_3c=1;
f_eta_3E=1;
f_eta_2E=1;
f_eta_1E=1;
EngineModel=6
hp=[100 176 316 562 1000 1780 3160 5620 10000];
ep=0.9
```

```
EngineModel =
     6
ep =
                        0.9
```

The following MATLAB cell needs to be executed several times (e.g., perhaps 5) until its calculated quantities stop changing. Only then are its set of calculated quantities self-consistent. As implemented here, the user must manually initiate each of those iterative executions and decide when to stop iterating. .

```
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);
R=GasConstant('MarsMix');
T0=215;T4=1290;p0=850;
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);

Ec1=1.27e5
Ec2=1.27e5
Ec3=1.27e5
EE3=2.5e5
EE2=2.95e5
EE1=2.95e5
display(' ')
etac1=0.85*f_eta_1c
etac2=0.85*f_eta_2c
etac3=0.85*f_eta_3c
etaE3=0.88*f_eta_3E
etaE2=0.88*f_eta_2E
etaE1=0.88*f_eta_1E
display(' ')
T1c_out_s=interp1(h,T,Ec1+h0);
T1c_out=interp1(h,T,Ec1/etac1+h0);
pic1=exp((interp1(T,S0,T1c_out_s)-S00)/R)
p1c_out=p0*pic1;
p2c_in=p1c_out;
T2c_in=T1c_out;
T2c_out_s=interp1(h,T,Ec2+Ec1/etac1+h0);
T2c_out=interp1(h,T,Ec2/etac2+Ec1/etac1+h0);
```

```
pic2=exp((interp1(T,S0,T2c_out_s)-interp1(T,S0,T2c_in))/R)
p2c_out=p2c_in*pic2;
p3c_in=p2c_out;
T3c_in=T2c_out;
T3c_out_s=interp1(h,T,Ec3+Ec2/etac2+Ec1/etac1+h0);
T3c_out=interp1(h,T,Ec3/etac3+Ec2/etac2+Ec1/etac1+h0);
pic3=exp((interp1(T,S0,T3c_out_s)-interp1(T,S0,T3c_in))/R)
p3c_out=p3c_in*pic3;
p3E_in=ep*p3c_out;
T3E_in=T4;
T3E_out_s=interp1(h,T,h4-EE3);
T3E_out=interp1(h,T,h4-etaE3*EE3);
piE3=exp((S04-interp1(T,S0,T3E_out_s))/R)
p3E_out=p3E_in./piE3;
p2E_in=p3E_out;
T2E_in=T3E_out;
T2E_out_s=interp1(h,T,h4-etaE3*EE3-EE2);
T2E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2);
piE2=exp((interp1(T,S0,T3E_out)-interp1(T,S0,T2E_out_s))/R)
p2E_out=p2E_in/piE2;
p1E_in=p2E_out;
T1E_in=T2E_out;
T1E_out_s=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-EE1);
T1E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-etaE1*EE1);
piE1=exp((interp1(T,S0,T2E_out)-interp1(T,S0,T1E_out_s))/R)
p1E_out=p1E_in/piE1;
display(' ')
Wsp=EE1*etaE1+EE2*etaE2+EE3*etaE3-Ec1/etac1-Ec2/etac2-Ec3/etac3
HeatAdded=h4-(Ec3+Ec2/etac2+Ec1/etac1+h0)
CycleEfficiency=Wsp/HeatAdded
mdot=746*hp(EngineModel)/Wsp
ReactorPower=HeatAdded*mdot
display(' ')
ShaftPower1c=-Ec1/etac1*mdot
ShaftPower2c=-Ec2/etac2*mdot
ShaftPower3c=-Ec3/etac3*mdot
ShaftPower3E=+EE3*etaE3*mdot
ShaftPower2E=+EE2*etaE2*mdot
ShaftPower1E=+EE1*etaE1*mdot
display(' ')
Q1c=mdot*R*T0/p0
Q2c=mdot*R*T2c_in/p2c_in
Q3c=mdot*R*T3c_in/p3c_in
Q3E=mdot*R*T3E_out./p3E_out
Q2E=mdot*R*T2E_out/p2E_out
Q1E=mdot*R*T1E_out/p1E_out
display(' ')
display('                    Pressures   Temperatures')
display([p0 T0; p2c_in T2c_in; p3c_in T3c_in; p3c_out T3c_out; p3E_in
T3E_in; p2E_in T2E_in; p1E_in T1E_in; p1E_out T1E_out])
display(' ')
NsE_ideal=0.55, DsE_ideal=3.3
Nsc_ideal=0.70,Dsc_ideal=3.7
D1c_ideal=Dsc_ideal*Q1c^0.50/Ec1^0.25;
D2c_ideal=Dsc_ideal*Q2c^0.50/Ec2^0.25;
```

```
D3c_ideal=Dsc_ideal*Q3c^0.50/Ec3^0.25;
D3E_ideal=DsE_ideal*Q3E^0.50/EE3^0.25;
D2E_ideal=DsE_ideal*Q2E^0.50/EE2^0.25;
D1E_ideal=DsE_ideal*Q1E^0.50/EE1^0.25;
display(' ')
N1c_ideal=Nsc_ideal*Q1c^-0.50*Ec1^0.75;
N2c_ideal=Nsc_ideal*Q2c^-0.50*Ec2^0.75;
N3c_ideal=Nsc_ideal*Q3c^-0.50*Ec3^0.75;
N3E_ideal=NsE_ideal*Q3E^-0.50*EE3^0.75;
N2E_ideal=NsE_ideal*Q2E^-0.50*EE2^0.75;
N1E_ideal=NsE_ideal*Q1E^-0.50*EE1^0.75;
display(' ')
N1=(N1c_ideal+N1E_ideal)/2
N2=(N2c_ideal+N2E_ideal)/2
N3=(N3c_ideal+N3E_ideal)/2
display(' ')
N1rpm=N1*30/pi
N2rpm=N2*30/pi
N3rpm=N3*30/pi
display(' ')
Ns1c=N1*Q1c^0.5*Ec1^-0.75
Ns2c=N2*Q2c^0.5*Ec2^-0.75
Ns3c=N3*Q3c^0.5*Ec3^-0.75
Ns3E=N3*Q3E^0.5*EE3^-0.75
Ns2E=N2*Q2E^0.5*EE2^-0.75
Ns1E=N1*Q1E^0.5*EE1^-0.75
display(' ')
Ds1c= 10^(log10(Dsc_ideal)-0.5*log10(Ns1c/Nsc_ideal))
Ds2c= 10^(log10(Dsc_ideal)-0.5*log10(Ns2c/Nsc_ideal))
Ds3c= 10^(log10(Dsc_ideal)-0.5*log10(Ns3c/Nsc_ideal))
Ds3E= 10^(log10(DsE_ideal)-1.0*log10(Ns3E/NsE_ideal))
Ds2E= 10^(log10(DsE_ideal)-1.0*log10(Ns2E/NsE_ideal))
Ds1E= 10^(log10(DsE_ideal)-1.0*log10(Ns1E/NsE_ideal))
display(' ')
D1c=Ds1c*Q1c^0.50/Ec1^0.25
D2c=Ds2c*Q2c^0.50/Ec2^0.25
D3c=Ds3c*Q3c^0.50/Ec3^0.25
D3E=Ds3E*Q3E^0.50/EE3^0.25
D2E=Ds2E*Q2E^0.50/EE2^0.25
D1E=Ds1E*Q1E^0.50/EE1^0.25
display(' ')
vtip1c=N1*D1c/2
vtip2c=N2*D2c/2
vtip3c=N3*D3c/2
vtip3E=N3*D3E/2
vtip2E=N2*D2E/2
vtip1E=N1*D1E/2
display(' ')
Re1c=vtip1c*D1c*p0/R/T0/Viscosity('MarsMix',T0)
Re2c=vtip2c*D2c*p2c_in/R/T2c_in/ Viscosity('MarsMix',T2c_in)
Re3c=vtip3c*D3c*p3c_in/R/T3c_in/ Viscosity('MarsMix',T3c_in)
Re3E=vtip3E*D3E*p3E_out/R/T3E_out/ Viscosity('MarsMix',T3E_out)
Re2E=vtip2E*D2E*p2E_out/R/T2E_out/ Viscosity('MarsMix',T2E_out)
Re1E=vtip1E*D1E*p1E_out/R/T1E_out/ Viscosity('MarsMix',T1E_out)
log10Re_val=[3 3.5 4 4.5 5 5.5 6 6.5 7];
```

```
f_eta_val=[0.2 0.34 0.5 0.65 0.8 0.915 0.97 0.99 1.];
display(' ')
f_eta_1c=interp1(log10Re_val,f_eta_val,log10(Re1c),'linear','extrap' )
f_eta_2c=interp1(log10Re_val,f_eta_val,log10(Re2c),'linear','extrap' )
f_eta_3c=interp1(log10Re_val,f_eta_val,log10(Re3c),'linear','extrap' )
f_eta_3E=interp1(log10Re_val,f_eta_val,log10(Re3E),'linear','extrap' )
f_eta_2E=interp1(log10Re_val,f_eta_val,log10(Re2E),'linear','extrap' )
f_eta_1E=interp1(log10Re_val,f_eta_val,log10(Re1E),'linear','extrap' )

Ec1 =
      127000
Ec2 =
      127000
Ec3 =
      127000
EE3 =
      250000
EE2 =
      295000
EE1 =
      295000

etac1 =
         0.839143508371895
etac2 =
         0.842904512347973
etac3 =
         0.844888530346554
etaE3 =
         0.858188189999536
etaE2 =
         0.844752807989222
etaE1 =
         0.816985095083911

pic1 =
          11.2851768923859
pic2 =
          4.07705643586425
pic3 =
          2.97893580982386
piE3 =
          3.00698371395171
piE2 =
          4.77706155642582
piE1 =
          7.35392445631477

Wsp =
          252429.751131342
HeatAdded =
          747275.187539701
CycleEfficiency =
          0.33780025797783
mdot =
```

```
          5.26039420491719
ReactorPower =
          3930962.06601225

ShaftPower1c =
         -796133.268456872
ShaftPower2c =
          -792580.95577579
ShaftPower3c =
         -790719.769565881
ShaftPower3E =
          1128602.04535048
ShaftPower2E =
          1310901.16884154
ShaftPower1E =
          1267810.77960652

Q1c =
           254.21925361709
Q2c =
          41.2773584832883
Q3c =
          14.0786478371988
Q3E =
          32.2664622274477
Q2E =
           125.96691109509
Q1E =
          714.917478233357

                  Pressures   Temperatures
ans =
                         850                        215
          9592.40035852798          393.958330786903
          39108.7576171231          547.830314241495
           116502.47854337          688.707177711846
          104852.230689033                      1290
          34869.5705276163          1119.46119610518
           7299.3764295777           914.85725326327
           992.58246028211          706.046990631361

NsE_ideal =
                       0.55
DsE_ideal =
                        3.3
Nsc_ideal =
                        0.7
Dsc_ideal =
                        3.7


N1 =
           277.86636445185
N2 =
           676.64174162124
```

```
N3 =
         1168.80607323384

N1rpm =
         2653.42832528917
N2rpm =
         6461.45267287977
N3rpm =
         11161.2758442596

Ns1c =
        0.658547958568321
Ns2c =
        0.646192545225004
Ns3c =
        0.651883729715059
Ns3E =
        0.593831359645808
Ns2E =
        0.599957537087517
Ns1E =
        0.586945022895646

Ds1c =
         3.81467058195226
Ds2c =
          3.8509667808052
Ds3c =
         3.83411975250959
Ds3E =
         3.05642329344573
Ds2E =
         3.02521409900255
Ds1E =
         3.09228280196643

D1c =
         3.22188718598429
D2c =
         1.31061396338778
D3c =
        0.762070615021515
D3E =
        0.776433337216615
D2E =
         1.45689700906486
D1E =
         3.54773896985017

vtip1c =
         447.627039521729
vtip2c =
         443.408057389912
vtip3c =
         445.356381535099
```

```
vtip3E =
                  453.75
vtip2E =
           492.89866478821
vtip1E =
           492.89866478821

Re1c =
           2695823.2811661
Re2c =
          3824884.90316403
Re3c =
          5004092.30911813
Re3E =
          1350038.96853235
Re2E =
           810220.79721054
Re1E =
           418549.52037084

f_eta_1c =
         0.987227656782634
f_eta_2c =
         0.991652367421569
f_eta_3c =
         0.993986506248129
f_eta_3E =
         0.975213852179039
f_eta_2E =
         0.959946372538061
f_eta_1E =
         0.928392153345262
```

Results Extracted From MATLAB Output:

Summary of Engine Model #6 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**1780 horsepower**					
Output Power (SI units)	**1.328 MW**					
Thermal Input Power (from Nuclear Reactor)	**3.931 MW**					
Thermal Conversion Cycle Energy Efficiency	**0.3378**					
Air Mass Flow Rate (kg/sec)	**5.260 kg/sec**					
	Low Pressure Stages		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**277.9 radians/sec**		**676.6 radians/sec**		**1168.8radians/sec**	
Shaft Speed (rpm)	**2,653.4 rpm**		**6,461.4 rpm**		**11,161.3 rpm**	
	Compressor	**Turbine**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**3.222 m**	**3.548 m**	**1.311 m**	**1.457 m**	**0.762 m**	**0.776 m**
Rotor Tip Speed (m/s)	**447.6 m/s**	**492.9 m/s**	**443.4 m/s**	**492.9 m/s**	**445.4 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.81**	**3.09**	**3.85**	**3.03**	**3.83**	**3.06**
Ns: Specific Speed	**0.659**	**0.587**	**0.646**	**0.600**	**0.652**	**0.594**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$2.70*10^6$**	**$4.19*10^5$**	**$3.82*10^6$**	**$8.10*10^5$**	**$5.00*10^6$**	**$1.35*10^6$**
Reynolds Number Efficiency Factor	**0.987**	**0.928**	**0.992**	**0.960**	**0.994**	**0.975**
Assumed Efficiency (from Balje*Reynolds)	**0.839**	**0.817**	**0.843**	**0.845**	**0.845**	**0.858**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**914.9 Kelvins**	**394.0 Kelvins**	**1119.5 Kelvins**	**547.8 Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,299 Pa**	**9,592 Pa**	**34,870 Pa**	**39,109 Pa**	**104,852 Pa**
Pressure Ratio (Calculated)	**11.285**	**7.354**	**4.077**	**4.777**	**2.979**	**3.007**
Discharge Air Temperature (Kelvins)	**394.0 Kelvins**	**706.0Kelvins**	**547.8 Kelvins**	**914.9 Kelvins**	**688.7 Kelvins**	**1119.5Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**993 Pa**	**39,109 Pa**	**7,299 Pa**	**116,502 Pa**	**34,870 Pa**
Q:Volumetric Flow Rate On Low Pressure Side	**254.22 m^3/s**	**714.92 m^3/s**	**41.28 m^3/s**	**125.97 m^3/s**	**14.08 m^3/s**	**32.27 m^3/s**
Shaft Powers (kW) [Total=1.328MW=1780hp]	**-0.796133 MW**	**+1.267811 MW**	**-0.792581 MW**	**+1.310901 MW**	**-0.790720 MW**	**+1.128602 MW**

Based on the results for engine models 1 through 6, models 7 through 9 cannot be implemented using single-unit radial flow turbomachine stages without rotor diameters exceeding 4 meters. To avoid such large diameter stages, air flows will be split between multiple stages operating in parallel for engine models 7 through 9. The following MATLAB script has been suitably modified.

First we initialize variables in preparation for an iterative set of calculations.

```
clear
f_eta_1c=1;
f_eta_2c=1;
f_eta_3c=1;
f_eta_3E=1;
f_eta_2E=1;
f_eta_1E=1;
EngineModel=7
hp=[100 176 316 562 1000 1780 3160 5620 10000];
ep=0.9

EngineModel =
     7
ep =
                        0.9
```

The following MATLAB cell needs to be executed several times (e.g., perhaps 5) until its calculated quantities stop changing. Only then are its set of calculated quantities self-consistent. As implemented here, the user must manually initiate each of those iterative executions and decide when to stop iterating. .

```
HorsePower=hp(EngineModel)
display(' ')
ParallelStages_c1=2
ParallelStages_c2=1
ParallelStages_c3=1
ParallelStages_E3=1
ParallelStages_E2=1
ParallelStages_E1=2
display(' ')
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);
R=GasConstant('MarsMix');
T0=215;T4=1290;p0=850;
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);

Ec1=1.27e5
Ec2=1.27e5
Ec3=1.27e5
EE3=2.5e5
EE2=2.95e5
```

```
EE1=2.95e5
display(' ')
etac1=0.85*f_eta_1c
etac2=0.85*f_eta_2c
etac3=0.85*f_eta_3c
etaE3=0.88*f_eta_3E
etaE2=0.88*f_eta_2E
etaE1=0.88*f_eta_1E
display(' ')
T1c_out_s=interp1(h,T,Ec1+h0);
T1c_out=interp1(h,T,Ec1/etac1+h0);
pic1=exp((interp1(T,S0,T1c_out_s)-S00)/R)
p1c_out=p0*pic1;
p2c_in=p1c_out;
T2c_in=T1c_out;
T2c_out_s=interp1(h,T,Ec2+Ec1/etac1+h0);
T2c_out=interp1(h,T,Ec2/etac2+Ec1/etac1+h0);
pic2=exp((interp1(T,S0,T2c_out_s)-interp1(T,S0,T2c_in))/R)
p2c_out=p2c_in*pic2;
p3c_in=p2c_out;
T3c_in=T2c_out;
T3c_out_s=interp1(h,T,Ec3+Ec2/etac2+Ec1/etac1+h0);
T3c_out=interp1(h,T,Ec3/etac3+Ec2/etac2+Ec1/etac1+h0);
pic3=exp((interp1(T,S0,T3c_out_s)-interp1(T,S0,T3c_in))/R)
p3c_out=p3c_in*pic3;
p3E_in=ep*p3c_out;
T3E_in=T4;
T3E_out_s=interp1(h,T,h4-EE3);
T3E_out=interp1(h,T,h4-etaE3*EE3);
piE3=exp((S04-interp1(T,S0,T3E_out_s))/R)
p3E_out=p3E_in./piE3;
p2E_in=p3E_out;
T2E_in=T3E_out;
T2E_out_s=interp1(h,T,h4-etaE3*EE3-EE2);
T2E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2);
piE2=exp((interp1(T,S0,T3E_out)-interp1(T,S0,T2E_out_s))/R)
p2E_out=p2E_in/piE2;
p1E_in=p2E_out;
T1E_in=T2E_out;
T1E_out_s=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-EE1);
T1E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-etaE1*EE1);
piE1=exp((interp1(T,S0,T2E_out)-interp1(T,S0,T1E_out_s))/R)
p1E_out=p1E_in/piE1;
display(' ')
Wsp=EE1*etaE1+EE2*etaE2+EE3*etaE3-Ec1/etac1-Ec2/etac2-Ec3/etac3
HeatAdded=h4-(Ec3+Ec2/etac2+Ec1/etac1+h0)
CycleEfficiency=Wsp/HeatAdded
mdot=746*hp(EngineModel)/Wsp
ReactorPower=HeatAdded*mdot
display(' ')
ShaftPower1c=-Ec1/etac1*mdot
ShaftPower2c=-Ec2/etac2*mdot
ShaftPower3c=-Ec3/etac3*mdot
ShaftPower3E=+EE3*etaE3*mdot
ShaftPower2E=+EE2*etaE2*mdot
```

```
ShaftPower1E=+EE1*etaE1*mdot
display(' ')
Q1c=mdot*R*T0/p0
Q2c=mdot*R*T2c_in/p2c_in
Q3c=mdot*R*T3c_in/p3c_in
Q3E=mdot*R*T3E_out./p3E_out
Q2E=mdot*R*T2E_out/p2E_out
Q1E=mdot*R*T1E_out/p1E_out
display(' ')
display('                    Pressures   Temperatures')
display([p0 T0; p2c_in T2c_in; p3c_in T3c_in; p3c_out T3c_out; p3E_in
T3E_in; p2E_in T2E_in; p1E_in T1E_in; p1E_out T1E_out])
display(' ')
NsE_ideal=0.55, DsE_ideal=3.3
Nsc_ideal=0.70,Dsc_ideal=3.7
D1c_ideal=Dsc_ideal* (Q1c/ParallelStages_c1)^0.50/Ec1^0.25;
D2c_ideal=Dsc_ideal* (Q2c/ParallelStages_c2)^0.50/Ec2^0.25;
D3c_ideal=Dsc_ideal* (Q3c/ParallelStages_c3)^0.50/Ec3^0.25;
D3E_ideal=DsE_ideal* (Q3E/ParallelStages_E3)^0.50/EE3^0.25;
D2E_ideal=DsE_ideal* (Q2E/ParallelStages_E2)^0.50/EE2^0.25;
D1E_ideal=DsE_ideal* (Q1E/ParallelStages_E1)^0.50/EE1^0.25;
display(' ')
N1c_ideal=Nsc_ideal*(Q1c/ParallelStages_c1)^-0.50*Ec1^0.75;
N2c_ideal=Nsc_ideal*(Q2c/ParallelStages_c2)^-0.50*Ec2^0.75;
N3c_ideal=Nsc_ideal*(Q3c/ParallelStages_c3)^-0.50*Ec3^0.75;
N3E_ideal=NsE_ideal*(Q3E/ParallelStages_E3)^-0.50*EE3^0.75;
N2E_ideal=NsE_ideal*(Q2E/ParallelStages_E2)^-0.50*EE2^0.75;
N1E_ideal=NsE_ideal*(Q1E/ParallelStages_E1)^-0.50*EE1^0.75;
display(' ')
N1=(N1c_ideal+N1E_ideal)/2
N2=(N2c_ideal+N2E_ideal)/2
N3=(N3c_ideal+N3E_ideal)/2
display(' ')
N1rpm=N1*30/pi
N2rpm=N2*30/pi
N3rpm=N3*30/pi
display(' ')
Ns1c=N1*(Q1c/ParallelStages_c1)^0.5*Ec1^-0.75
Ns2c=N2*(Q2c/ParallelStages_c2)^0.5*Ec2^-0.75
Ns3c=N3*(Q3c/ParallelStages_c3)^0.5*Ec3^-0.75
Ns3E=N3*(Q3E/ParallelStages_E3)^0.5*EE3^-0.75
Ns2E=N2*(Q2E/ParallelStages_E2)^0.5*EE2^-0.75
Ns1E=N1*(Q1E/ParallelStages_E1)^0.5*EE1^-0.75
display(' ')
Ds1c= 10^(log10(Dsc_ideal)-0.5*log10(Ns1c/Nsc_ideal))
Ds2c= 10^(log10(Dsc_ideal)-0.5*log10(Ns2c/Nsc_ideal))
Ds3c= 10^(log10(Dsc_ideal)-0.5*log10(Ns3c/Nsc_ideal))
Ds3E= 10^(log10(DsE_ideal)-1.0*log10(Ns3E/NsE_ideal))
Ds2E= 10^(log10(DsE_ideal)-1.0*log10(Ns2E/NsE_ideal))
Ds1E= 10^(log10(DsE_ideal)-1.0*log10(Ns1E/NsE_ideal))
display(' ')
D1c=Ds1c*(Q1c/ParallelStages_c1)^0.50/Ec1^0.25
D2c=Ds2c*(Q2c/ParallelStages_c2)^0.50/Ec2^0.25
D3c=Ds3c*(Q3c/ParallelStages_c3)^0.50/Ec3^0.25
D3E=Ds3E*(Q3E/ParallelStages_E3)^0.50/EE3^0.25
```

```
D2E=Ds2E*(Q2E/ParallelStages_E2)^0.50/EE2^0.25
D1E=Ds1E*(Q1E/ParallelStages_E1)^0.50/EE1^0.25
display(' ')
vtip1c=N1*D1c/2
vtip2c=N2*D2c/2
vtip3c=N3*D3c/2
vtip3E=N3*D3E/2
vtip2E=N2*D2E/2
vtip1E=N1*D1E/2
display(' ')
Re1c=vtip1c*D1c*p0/R/T0/Viscosity('MarsMix',T0)
Re2c=vtip2c*D2c*p2c_in/R/T2c_in/ Viscosity('MarsMix',T2c_in)
Re3c=vtip3c*D3c*p3c_in/R/T3c_in/ Viscosity('MarsMix',T3c_in)
Re3E=vtip3E*D3E*p3E_out/R/T3E_out/ Viscosity('MarsMix',T3E_out)
Re2E=vtip2E*D2E*p2E_out/R/T2E_out/ Viscosity('MarsMix',T2E_out)
Re1E=vtip1E*D1E*p1E_out/R/T1E_out/ Viscosity('MarsMix',T1E_out)
log10Re_val=[3 3.5 4 4.5 5 5.5 6 6.5 7];
f_eta_val=[0.2 0.34 0.5 0.65 0.8 0.915 0.97 0.99 1.];
display(' ')
f_eta_1c=interp1(log10Re_val,f_eta_val,log10(Re1c),'linear','extrap' )
f_eta_2c=interp1(log10Re_val,f_eta_val,log10(Re2c),'linear','extrap' )
f_eta_3c=interp1(log10Re_val,f_eta_val,log10(Re3c),'linear','extrap' )
f_eta_3E=interp1(log10Re_val,f_eta_val,log10(Re3E),'linear','extrap' )
f_eta_2E=interp1(log10Re_val,f_eta_val,log10(Re2E),'linear','extrap' )
f_eta_1E=interp1(log10Re_val,f_eta_val,log10(Re1E),'linear','extrap' )

HorsePower =
        3160

ParallelStages_c1 =
     2
ParallelStages_c2 =
     1
ParallelStages_c3 =
     1
ParallelStages_E3 =
     1
ParallelStages_E2 =
     1
ParallelStages_E1 =
     2

Ec1 =
      127000
Ec2 =
      127000
Ec3 =
      127000
EE3 =
      250000
EE2 =
      295000
EE1 =
      295000
```

```
etac1 =
         0.838155735080374
etac2 =
         0.844964501791943
etac3 =
         0.846954406174138
etaE3 =
         0.862478298213646
etaE2 =
         0.854722744268322
etaE1 =
         0.814082195266261

pic1 =
          11.2851768923859
pic2 =
          4.07465411032471
pic3 =
          2.97978687689905
piE3 =
          3.00698371395171
piE2 =
          4.78392191024364
piE1 =
          7.42040473042535

Wsp =
          256142.666353201
HeatAdded =
          747464.152653873
CycleEfficiency =
         0.342682208161777
mdot =
          9.20330858408956
ReactorPower =
          6879143.25241862

ShaftPower1c =
         -1394514.33815852
ShaftPower2c =
         -1383277.27105768
ShaftPower3c =
          -1380027.2855999
ShaftPower3E =
          1984413.48138515
ShaftPower2E =
          2320551.76495566
ShaftPower1E =
          2210213.64847529

Q1c =
          444.768613893612
Q2c =
          72.2520431629858
Q3c =
```

```
          24.6375138525691
Q3E =
          56.4252670630461
Q2E =
          219.955381613777
Q1E =
          1259.19527917193

                   Pressures   Temperatures
ans =
                         850                        215
          9592.40035852798          394.151179802023
          39085.7135487562          547.646514367144
          116467.096306819          688.203480299337
          104820.386676137                      1290
          34858.9805092043          1118.59714445294
          7286.69513491891          911.487240233818
          981.980821752457          703.204262165814

NsE_ideal =
                        0.55
DsE_ideal =
                         3.3
Nsc_ideal =
                         0.7
Dsc_ideal =
                         3.7


N1 =
          296.624270620431
N2 =
          511.720375933409
N3 =
          883.684274444711

N1rpm =
          2832.55313461618
N2rpm =
           4886.5696386387
N3rpm =
          8438.56322462705

Ns1c =
         0.657515448454999
Ns2c =
         0.646554272473139
Ns3c =
         0.651992765173565
Ns3E =
         0.593716244558647
Ns2E =
         0.599561164461787
Ns1E =
         0.587992405694002
```

```
Ds1c =
          3.81766453511865
Ds2c =
          3.84988938130044
Ds3c =
          3.83379914121814
Ds3E =
          3.05701590049843
Ds2E =
          3.02721408186817
Ds1E =
          3.08677456107239

D1c =
          3.01577475755291
D2c =
          1.73349417733344
D3c =
            1.008037912376
D3E =
          1.02695049153189
D2E =
          1.92643751536816
D1E =
          3.32338728558688

vtip1c =
          447.275993907319
vtip2c =
          443.532146051721
vtip3c =
          445.393625605374
vtip3E =
                    453.75
vtip2E =
           492.89866478821
vtip1E =
           492.89866478821

Re1c =
          2521385.60136318
Re2c =
           5055850.5040288
Re3c =
          6619838.93148812
Re3E =
          1787410.81717609
Re2E =
          1076207.82300132
Re1E =
          390623.364195161

f_eta_1c =
         0.986065570736827
```

```
f_eta_2c =
         0.994075884477024
f_eta_3c =
         0.996416948453395
f_eta_3E =
         0.980088975275998
f_eta_2E =
         0.971275845785095
f_eta_1E =
          0.92509340376072
```

Results Extracted From MATLAB Output:

Summary of Engine Model #7 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**3160 horsepower**					
Output Power (SI units)	**2.357 MW**					
Thermal Input Power (from Nuclear Reactor)	**6.879 MW**					
Thermal Conversion Cycle Energy Efficiency	**0.3427**					
Air Mass Flow Rate (kg/sec)	**9.203 kg/sec**					
	Low Pressure Stages Operating In Parallel		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**296.6 radians/sec**		**511.7radians/sec**		**883.7 radians/sec**	
Shaft Speed (rpm)	**2,832.6 rpm**		**4,886.6 rpm**		**8,438.6 rpm**	
	Compressors (2)	**Turbines (2)**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**3.016 m**	**3.323 m**	**1.733 m**	**1.926 m**	**1.008 m**	**1.027 m**
Rotor Tip Speed (m/s)	**447.3 m/s**	**492.9 m/s**	**443.5 m/s**	**492.9 m/s**	**445.4 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.09**	**3.85**	**3.03**	**3.84**	**3.06**
Ns: Specific Speed	**0.658**	**0.588**	**0.647**	**0.600**	**0.652**	**0.594**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$2.52*10^6$**	**$3.91*10^5$**	**$5.06*10^6$**	**$1.08*10^6$**	**$6.62*10^6$**	**$1.79*10^6$**
Reynolds Number Efficiency Factor	**0.986**	**0.925**	**0.994**	**0.971**	**0.996**	**0.980**
Assumed Efficiency (from Balje*Reynolds)	**0.838**	**0.814**	**0.845**	**0.855**	**0.847**	**0.862**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**911.5 Kelvins**	**394.2 Kelvins**	**1118.6 Kelvins**	**547.6 Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,287 Pa**	**9,592 Pa**	**34,859 Pa**	**39,086 Pa**	**104,820 Pa**
Pressure Ratio (Calculated)	**11.285**	**7.420**	**4.075**	**4.784**	**2.980**	**3.007**
Discharge Air Temperature (Kelvins)	**394.2 Kelvins**	**703.2 Kelvins**	**547.6 Kelvins**	**911.5 Kelvins**	**688.2 Kelvins**	**1118.6 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**982 Pa**	**39,086 Pa**	**7,287 Pa**	**116,467 Pa**	**34,859 Pa**
Q:Volumetric Flow Rate Total Low Pressure Sides	**444.77 m^3/s**	**1,259.20 m^3/s**	**72.25 m^3/s**	**219.96 m^3/s**	**24.64 m^3/s**	**56.43 m^3/s**
Shaft Powers (kW) [Total=2.357MW=3160hp]	**-1.395 MW**	**+2.210 MW**	**-1.383 MW**	**+2.321 MW**	**-1.380MW**	**+1.984 MW**

As with engine model number 7, the higher power models will some split air flows between multiple stages operating in parallel in order to obtain efficient turbomachine parameters without exceeding 4 meters for turbomachine diameters. The following MATLAB script has been suitably modified.

First we initialize variables in preparation for an iterative set of calculations.

```
clear
f_eta_1c=1;
f_eta_2c=1;
f_eta_3c=1;
f_eta_3E=1;
f_eta_2E=1;
f_eta_1E=1;
EngineModel=8
hp=[100 176 316 562 1000 1780 3160 5620 10000];
ep=0.9

EngineModel =
     8
ep =
                        0.9
```

The following MATLAB cell needs to be executed several times (e.g., perhaps 5) until its calculated quantities stop changing. Only then are its set of calculated quantities self-consistent. As implemented here, the user must manually initiate each of those iterative executions and decide when to stop iterating. .

```
HorsePower=hp(EngineModel)
display(' ')
ParallelStages_c1=4
ParallelStages_c2=1
ParallelStages_c3=1
ParallelStages_E3=1
ParallelStages_E2=1
ParallelStages_E1=4
display(' ')
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);
R=GasConstant('MarsMix');
T0=215;T4=1290;p0=850;
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);

Ec1=1.27e5
Ec2=1.27e5
Ec3=1.27e5
EE3=2.5e5
```

```
EE2=2.95e5
EE1=2.95e5
display(' ')
etac1=0.85*f_eta_1c
etac2=0.85*f_eta_2c
etac3=0.85*f_eta_3c
etaE3=0.88*f_eta_3E
etaE2=0.88*f_eta_2E
etaE1=0.88*f_eta_1E
display(' ')
T1c_out_s=interp1(h,T,Ec1+h0);
T1c_out=interp1(h,T,Ec1/etac1+h0);
pic1=exp((interp1(T,S0,T1c_out_s)-S00)/R)
p1c_out=p0*pic1;
p2c_in=p1c_out;
T2c_in=T1c_out;
T2c_out_s=interp1(h,T,Ec2+Ec1/etac1+h0);
T2c_out=interp1(h,T,Ec2/etac2+Ec1/etac1+h0);
pic2=exp((interp1(T,S0,T2c_out_s)-interp1(T,S0,T2c_in))/R)
p2c_out=p2c_in*pic2;
p3c_in=p2c_out;
T3c_in=T2c_out;
T3c_out_s=interp1(h,T,Ec3+Ec2/etac2+Ec1/etac1+h0);
T3c_out=interp1(h,T,Ec3/etac3+Ec2/etac2+Ec1/etac1+h0);
pic3=exp((interp1(T,S0,T3c_out_s)-interp1(T,S0,T3c_in))/R)
p3c_out=p3c_in*pic3;
p3E_in=ep*p3c_out;
T3E_in=T4;
T3E_out_s=interp1(h,T,h4-EE3);
T3E_out=interp1(h,T,h4-etaE3*EE3);
piE3=exp((S04-interp1(T,S0,T3E_out_s))/R)
p3E_out=p3E_in./piE3;
p2E_in=p3E_out;
T2E_in=T3E_out;
T2E_out_s=interp1(h,T,h4-etaE3*EE3-EE2);
T2E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2);
piE2=exp((interp1(T,S0,T3E_out)-interp1(T,S0,T2E_out_s))/R)
p2E_out=p2E_in/piE2;
p1E_in=p2E_out;
T1E_in=T2E_out;
T1E_out_s=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-EE1);
T1E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-etaE1*EE1);
piE1=exp((interp1(T,S0,T2E_out)-interp1(T,S0,T1E_out_s))/R)
p1E_out=p1E_in/piE1;
display(' ')
Wsp=EE1*etaE1+EE2*etaE2+EE3*etaE3-Ec1/etac1-Ec2/etac2-Ec3/etac3
HeatAdded=h4-(Ec3+Ec2/etac2+Ec1/etac1+h0)
CycleEfficiency=Wsp/HeatAdded
mdot=746*hp(EngineModel)/Wsp
ReactorPower=HeatAdded*mdot
display(' ')
ShaftPower1c=-Ec1/etac1*mdot
ShaftPower2c=-Ec2/etac2*mdot
ShaftPower3c=-Ec3/etac3*mdot
ShaftPower3E=+EE3*etaE3*mdot
```

```
ShaftPower2E=+EE2*etaE2*mdot
ShaftPower1E=+EE1*etaE1*mdot
display(' ')
Q1c=mdot*R*T0/p0
Q2c=mdot*R*T2c_in/p2c_in
Q3c=mdot*R*T3c_in/p3c_in
Q3E=mdot*R*T3E_out./p3E_out
Q2E=mdot*R*T2E_out/p2E_out
Q1E=mdot*R*T1E_out/p1E_out
display(' ')
display('                    Pressures   Temperatures')
display([p0 T0; p2c_in T2c_in; p3c_in T3c_in; p3c_out T3c_out; p3E_in
T3E_in; p2E_in T2E_in; p1E_in T1E_in; p1E_out T1E_out])
display(' ')
NsE_ideal=0.55, DsE_ideal=3.3

D1c_ideal=Dsc_ideal* (Q1c/ParallelStages_c1)^0.50/Ec1^0.25;
D2c_ideal=Dsc_ideal* (Q2c/ParallelStages_c2)^0.50/Ec2^0.25;
D3c_ideal=Dsc_ideal* (Q3c/ParallelStages_c3)^0.50/Ec3^0.25;
D3E_ideal=DsE_ideal* (Q3E/ParallelStages_E3)^0.50/EE3^0.25;
D2E_ideal=DsE_ideal* (Q2E/ParallelStages_E2)^0.50/EE2^0.25;
D1E_ideal=DsE_ideal* (Q1E/ParallelStages_E1)^0.50/EE1^0.25;
display(' ')
N1c_ideal=Nsc_ideal*(Q1c/ParallelStages_c1)^-0.50*Ec1^0.75;
N2c_ideal=Nsc_ideal*(Q2c/ParallelStages_c2)^-0.50*Ec2^0.75;
N3c_ideal=Nsc_ideal*(Q3c/ParallelStages_c3)^-0.50*Ec3^0.75;
N3E_ideal=NsE_ideal*(Q3E/ParallelStages_E3)^-0.50*EE3^0.75;
N2E_ideal=NsE_ideal*(Q2E/ParallelStages_E2)^-0.50*EE2^0.75;
N1E_ideal=NsE_ideal*(Q1E/ParallelStages_E1)^-0.50*EE1^0.75;
display(' ')
N1=(N1c_ideal+N1E_ideal)/2
N2=(N2c_ideal+N2E_ideal)/2
N3=(N3c_ideal+N3E_ideal)/2
display(' ')
N1rpm=N1*30/pi
N2rpm=N2*30/pi
N3rpm=N3*30/pi
display(' ')
Ns1c=N1*(Q1c/ParallelStages_c1)^0.5*Ec1^-0.75
Ns2c=N2*(Q2c/ParallelStages_c2)^0.5*Ec2^-0.75
Ns3c=N3*(Q3c/ParallelStages_c3)^0.5*Ec3^-0.75
Ns3E=N3*(Q3E/ParallelStages_E3)^0.5*EE3^-0.75
Ns2E=N2*(Q2E/ParallelStages_E2)^0.5*EE2^-0.75
Ns1E=N1*(Q1E/ParallelStages_E1)^0.5*EE1^-0.75
display(' ')
Ds1c= 10^(log10(Dsc_ideal)-0.5*log10(Ns1c/Nsc_ideal))
Ds2c= 10^(log10(Dsc_ideal)-0.5*log10(Ns2c/Nsc_ideal))
Ds3c= 10^(log10(Dsc_ideal)-0.5*log10(Ns3c/Nsc_ideal))
Ds3E= 10^(log10(DsE_ideal)-1.0*log10(Ns3E/NsE_ideal))
Ds2E= 10^(log10(DsE_ideal)-1.0*log10(Ns2E/NsE_ideal))
Ds1E= 10^(log10(DsE_ideal)-1.0*log10(Ns1E/NsE_ideal))
display(' ')
D1c=Ds1c*(Q1c/ParallelStages_c1)^0.50/Ec1^0.25
D2c=Ds2c*(Q2c/ParallelStages_c2)^0.50/Ec2^0.25
D3c=Ds3c*(Q3c/ParallelStages_c3)^0.50/Ec3^0.25
```

```
D3E=Ds3E*(Q3E/ParallelStages_E3)^0.50/EE3^0.25
D2E=Ds2E*(Q2E/ParallelStages_E2)^0.50/EE2^0.25
D1E=Ds1E*(Q1E/ParallelStages_E1)^0.50/EE1^0.25
display(' ')
vtip1c=N1*D1c/2
vtip2c=N2*D2c/2
vtip3c=N3*D3c/2
vtip3E=N3*D3E/2
vtip2E=N2*D2E/2
vtip1E=N1*D1E/2
display(' ')
Re1c=vtip1c*D1c*p0/R/T0/Viscosity('MarsMix',T0)
Re2c=vtip2c*D2c*p2c_in/R/T2c_in/ Viscosity('MarsMix',T2c_in)
Re3c=vtip3c*D3c*p3c_in/R/T3c_in/ Viscosity('MarsMix',T3c_in)
Re3E=vtip3E*D3E*p3E_out/R/T3E_out/ Viscosity('MarsMix',T3E_out)
Re2E=vtip2E*D2E*p2E_out/R/T2E_out/ Viscosity('MarsMix',T2E_out)
Re1E=vtip1E*D1E*p1E_out/R/T1E_out/ Viscosity('MarsMix',T1E_out)
log10Re_val=[3 3.5 4 4.5 5 5.5 6 6.5 7];
f_eta_val=[0.2 0.34 0.5 0.65 0.8 0.915 0.97 0.99 1.];
display(' ')
f_eta_1c=interp1(log10Re_val,f_eta_val,log10(Re1c),'linear','extrap' )
f_eta_2c=interp1(log10Re_val,f_eta_val,log10(Re2c),'linear','extrap' )
f_eta_3c=interp1(log10Re_val,f_eta_val,log10(Re3c),'linear','extrap' )
f_eta_3E=interp1(log10Re_val,f_eta_val,log10(Re3E),'linear','extrap' )
f_eta_2E=interp1(log10Re_val,f_eta_val,log10(Re2E),'linear','extrap' )
f_eta_1E=interp1(log10Re_val,f_eta_val,log10(Re1E),'linear','extrap' )

HorsePower =
        5620

ParallelStages_c1 =
     4
ParallelStages_c2 =
     1
ParallelStages_c3 =
     1
ParallelStages_E3 =
     1
ParallelStages_E2 =
     1
ParallelStages_E1 =
     4

Ec1 =
      127000
Ec2 =
      127000
Ec3 =
      127000
EE3 =
      250000
EE2 =
      295000
EE1 =
      295000
```

```
etac1 =
         0.837227615697022
etac2 =
         0.847054605712932
etac3 =
         0.849050477141037
etaE3 =
         0.866830850373512
etaE2 =
         0.859091069186838
etaE1 =
         0.811339200332744

pic1 =
          11.2851768923859
pic2 =
          4.07239899741081
pic3 =
          2.98070096195119
piE3 =
          3.00698371395171
piE2 =
          4.79090580119403
piE1 =
          7.46033175751621

Wsp =
          258283.356566013
HeatAdded =
          747667.049744782
CycleEfficiency =
         0.345452373023766
mdot =
          16.2322499434006
ReactorPower =
          12136318.4259023

ShaftPower1c =
         -2462288.27640332
ShaftPower2c =
         -2433722.37032677
ShaftPower3c =
         -2428001.39486812
ShaftPower3E =
          3517653.75547834
ShaftPower2E =
          4113769.38295929
ShaftPower1E =
          3885108.90316057

Q1c =
          784.456507324118
Q2c =
          127.492577530086
```

```
Q3c =
          43.4625810576784
Q3E =
          99.4660907807221
Q2E =
          387.755124469199
Q1E =
          2232.16250085249

                    Pressures   Temperatures
ans =
                         850                         215
          9592.40035852798          394.332763434182
          39064.0816028324          547.449163337677
          116438.345611302          687.683810578608
          104794.511050172                      1290
            34850.37533258          1117.72038993164
          7274.27688598976          909.490230119685
           975.06077778123           701.79158966943

NsE_ideal =
                       0.55
DsE_ideal =
                        3.3
Nsc_ideal =
                        0.7
Dsc_ideal =
                        3.7


N1 =
          315.493682815947
N2 =
           385.30955720101
N3 =
          665.443642219523

N1rpm =
          3012.74274806547
N2rpm =
           3679.4352389454
N3rpm =
           6354.5187004987

Ns1c =
         0.656738257485364
Ns2c =
         0.646694907950041
Ns3c =
         0.652103072082845
Ns3E =
           0.5935998716809
Ns2E =
         0.599407320182951
Ns1E =
```

```
         0.588785442967096

Ds1c =
          3.81992279940899
Ds2c =
          3.84947074458189
Ds3c =
          3.83347487314042
Ds3E =
          3.05761521622378
Ds2E =
          3.02799104863455
Ds1E =
          3.08261697309223

D1c =
          2.83372756641358
D2c =
          2.30246238466291
D3c =
          1.33874988818068
D3E =
          1.36375185278369
D2E =
          2.55845543188057
D1E =
          3.12461828324948

vtip1c =
          447.011573012446
vtip2c =
          443.580380953223
vtip3c =
          445.431300805965
vtip3E =
                    453.75
vtip2E =
           492.89866478821
vtip1E =
           492.89866478821

Re1c =
          2367781.58851641
Re2c =
          6710293.45526237
Re3c =
          8793174.96503594
Re3E =
          2376164.10916063
Re2E =
          1432175.64601112
Re1E =
          365949.780715219

f_eta_1c =
```

```
          0.984973665569386
f_eta_2c =
          0.996534830263929
f_eta_3c =
          0.998882914295173
f_eta_3E =
          0.985035057271249
f_eta_2E =
          0.976239851372937
f_eta_1E =
          0.921976364038151
```

Results Extracted From MATLAB Output:

Summary of Engine Model #8 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**5620 horsepower**					
Output Power (SI units)	**4.193 MW**					
Thermal Input Power (from Nuclear Reactor)	**12.136 MW**					
Thermal Conversion Cycle Energy Efficiency	**0.3455**					
Air Mass Flow Rate (kg/sec)	**16.232 kg/sec**					
	Low Pressure Stages Operating In Parallel		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**315.5 radians/sec**		**385.3 radians/sec**		**665.4 radians/sec**	
Shaft Speed (rpm)	**3,012.7 rpm**		**3,679.4 rpm**		**6,354.5 rpm**	
	Compressors (4)	**Turbines (4)**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**2.834 m**	**3.125 m**	**2.302 m**	**2.558 m**	**1.339 m**	**1.364 m**
Rotor Tip Speed (m/s)	**447.0 m/s**	**492.9 m/s**	**443.6 m/s**	**492.9 m/s**	**445.4 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.08**	**3.85**	**3.03**	**3.84**	**3.06**
Ns: Specific Speed	**0.657**	**0.589**	**0.647**	**0.599**	**0.652**	**0.594**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$2.37*10^6$**	**$3.66*10^5$**	**$6.71*10^6$**	**$1.43*10^6$**	**$8.79*10^6$**	**$2.38*10^6$**
Reynolds Number Efficiency Factor	**0.985**	**0.922**	**0.997**	**0.976**	**0.999**	**0.985**
Assumed Efficiency (from Balje*Reynolds)	**0.837**	**0.811**	**0.847**	**0.859**	**0.849**	**0.867**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**909.5 Kelvins**	**394.3 Kelvins**	**1117.7 Kelvins**	**547.4 Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,274 Pa**	**9,592 Pa**	**34,850 Pa**	**39,064 Pa**	**104,795 Pa**
Pressure Ratio (Calculated)	**11.285**	**7.460**	**4.072**	**4.791**	**2.981**	**3.007**
Discharge Air Temperature (Kelvins)	**394.3 Kelvins**	**701.8 Kelvins**	**547.4 Kelvins**	**909.5 Kelvins**	**687.7 Kelvins**	**1117.7 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**975 Pa**	**39,064 Pa**	**7,274 Pa**	**116,438 Pa**	**34,850 Pa**
Q:Volumetric Flow Rate Total Low Pressure Sides	**784.46 m^3/s**	**2232.16 m^3/s**	**127.49 m^3/s**	**387.76 m^3/s**	**43.36 m^3/s**	**99.47 m^3/s**
Shaft Powers (kW) [Total=4.193MW=5,620hp]	**-2.462 MW**	**+3.885 MW**	**-2.434 MW**	**+4.114 MW**	**-2.428 MW**	**+3.518 MW**

As with engine model number 7, the higher power models will some split air flows between multiple stages operating in parallel in order to obtain efficient turbomachine parameters without exceeding 4 meters for turbomachine diameters. The following MATLAB script has been suitably modified.

First we initialize variables in preparation for an iterative set of calculations.

```
clear
f_eta_1c=1;
f_eta_2c=1;
f_eta_3c=1;
f_eta_3E=1;
f_eta_2E=1;
f_eta_1E=1;
EngineModel=9
hp=[100 176 316 562 1000 1780 3160 5620 10000];
ep=0.9

EngineModel =
     9
ep =
                        0.9
```

The following MATLAB cell needs to be executed several times (e.g., perhaps 5) until its calculated quantities stop changing. Only then are its set of calculated quantities self-consistent. As implemented here, the user must manually initiate each of those iterative executions and decide when to stop iterating. .

```
HorsePower=hp(EngineModel)
display(' ')
ParallelStages_c1=6
ParallelStages_c2=1
ParallelStages_c3=1
ParallelStages_E3=1
ParallelStages_E2=1
ParallelStages_E1=6
display(' ')
T=140:1650;h=Enthalpy('MarsMix',T);S0=EntropyIntegral('MarsMix',T);
R=GasConstant('MarsMix');
T0=215;T4=1290;p0=850;
h0=Enthalpy('MarsMix',T0);
h4=Enthalpy('MarsMix',T4);
S00=EntropyIntegral('MarsMix',T0);
S04=EntropyIntegral('MarsMix',T4);

Ec1=1.27e5
Ec2=1.27e5
Ec3=1.27e5
EE3=2.5e5
EE2=2.95e5
EE1=2.95e5
```

```
display(' ')
etac1=0.85*f_eta_1c
etac2=0.85*f_eta_2c
etac3=0.85*f_eta_3c
etaE3=0.88*f_eta_3E
etaE2=0.88*f_eta_2E
etaE1=0.88*f_eta_1E
display(' ')
T1c_out_s=interp1(h,T,Ec1+h0);
T1c_out=interp1(h,T,Ec1/etac1+h0);
pic1=exp((interp1(T,S0,T1c_out_s)-S00)/R)
p1c_out=p0*pic1;
p2c_in=p1c_out;
T2c_in=T1c_out;
T2c_out_s=interp1(h,T,Ec2+Ec1/etac1+h0);
T2c_out=interp1(h,T,Ec2/etac2+Ec1/etac1+h0);
pic2=exp((interp1(T,S0,T2c_out_s)-interp1(T,S0,T2c_in))/R)
p2c_out=p2c_in*pic2;
p3c_in=p2c_out;
T3c_in=T2c_out;
T3c_out_s=interp1(h,T,Ec3+Ec2/etac2+Ec1/etac1+h0);
T3c_out=interp1(h,T,Ec3/etac3+Ec2/etac2+Ec1/etac1+h0);
pic3=exp((interp1(T,S0,T3c_out_s)-interp1(T,S0,T3c_in))/R)
p3c_out=p3c_in*pic3;
p3E_in=ep*p3c_out;
T3E_in=T4;
T3E_out_s=interp1(h,T,h4-EE3);
T3E_out=interp1(h,T,h4-etaE3*EE3);
piE3=exp((S04-interp1(T,S0,T3E_out_s))/R)
p3E_out=p3E_in./piE3;
p2E_in=p3E_out;
T2E_in=T3E_out;
T2E_out_s=interp1(h,T,h4-etaE3*EE3-EE2);
T2E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2);
piE2=exp((interp1(T,S0,T3E_out)-interp1(T,S0,T2E_out_s))/R)
p2E_out=p2E_in/piE2;
p1E_in=p2E_out;
T1E_in=T2E_out;
T1E_out_s=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-EE1);
T1E_out=interp1(h,T,h4-etaE3*EE3-etaE2*EE2-etaE1*EE1);
piE1=exp((interp1(T,S0,T2E_out)-interp1(T,S0,T1E_out_s))/R)
p1E_out=p1E_in/piE1;
display(' ')
Wsp=EE1*etaE1+EE2*etaE2+EE3*etaE3-Ec1/etac1-Ec2/etac2-Ec3/etac3
HeatAdded=h4-(Ec3+Ec2/etac2+Ec1/etac1+h0)
CycleEfficiency=Wsp/HeatAdded
mdot=746*hp(EngineModel)/Wsp
ReactorPower=HeatAdded*mdot
display(' ')
ShaftPower1c=-Ec1/etac1*mdot
ShaftPower2c=-Ec2/etac2*mdot
ShaftPower3c=-Ec3/etac3*mdot
ShaftPower3E=+EE3*etaE3*mdot
ShaftPower2E=+EE2*etaE2*mdot
ShaftPower1E=+EE1*etaE1*mdot
```

```
display(' ')
Q1c=mdot*R*T0/p0
Q2c=mdot*R*T2c_in/p2c_in
Q3c=mdot*R*T3c_in/p3c_in
Q3E=mdot*R*T3E_out./p3E_out
Q2E=mdot*R*T2E_out/p2E_out
Q1E=mdot*R*T1E_out/p1E_out
display(' ')
display('                    Pressures   Temperatures')
display([p0 T0; p2c_in T2c_in; p3c_in T3c_in; p3c_out T3c_out; p3E_in
T3E_in; p2E_in T2E_in; p1E_in T1E_in; p1E_out T1E_out])
display(' ')
NsE_ideal=0.55, DsE_ideal=3.3
Nsc_ideal=0.70,Dsc_ideal=3.7
D1c_ideal=Dsc_ideal* (Q1c/ParallelStages_c1)^0.50/Ec1^0.25;
D2c_ideal=Dsc_ideal* (Q2c/ParallelStages_c2)^0.50/Ec2^0.25;
D3c_ideal=Dsc_ideal* (Q3c/ParallelStages_c3)^0.50/Ec3^0.25;
D3E_ideal=DsE_ideal* (Q3E/ParallelStages_E3)^0.50/EE3^0.25;
D2E_ideal=DsE_ideal* (Q2E/ParallelStages_E2)^0.50/EE2^0.25;
D1E_ideal=DsE_ideal* (Q1E/ParallelStages_E1)^0.50/EE1^0.25;
display(' ')
N1c_ideal=Nsc_ideal*(Q1c/ParallelStages_c1)^-0.50*Ec1^0.75;
N2c_ideal=Nsc_ideal*(Q2c/ParallelStages_c2)^-0.50*Ec2^0.75;
N3c_ideal=Nsc_ideal*(Q3c/ParallelStages_c3)^-0.50*Ec3^0.75;
N3E_ideal=NsE_ideal*(Q3E/ParallelStages_E3)^-0.50*EE3^0.75;
N2E_ideal=NsE_ideal*(Q2E/ParallelStages_E2)^-0.50*EE2^0.75;
N1E_ideal=NsE_ideal*(Q1E/ParallelStages_E1)^-0.50*EE1^0.75;
display(' ')
N1=(N1c_ideal+N1E_ideal)/2;
N2=(N2c_ideal+N2E_ideal)/2;
N1=(N1+N2)/2
N2=N1
N3=(N3c_ideal+N3E_ideal)/2
display(' ')
N1rpm=N1*30/pi
N2rpm=N2*30/pi
N3rpm=N3*30/pi
display(' ')
Ns1c=N1*(Q1c/ParallelStages_c1)^0.5*Ec1^-0.75
Ns2c=N2*(Q2c/ParallelStages_c2)^0.5*Ec2^-0.75
Ns3c=N3*(Q3c/ParallelStages_c3)^0.5*Ec3^-0.75
Ns3E=N3*(Q3E/ParallelStages_E3)^0.5*EE3^-0.75
Ns2E=N2*(Q2E/ParallelStages_E2)^0.5*EE2^-0.75
Ns1E=N1*(Q1E/ParallelStages_E1)^0.5*EE1^-0.75
display(' ')
Ds1c= 10^(log10(Dsc_ideal)-0.5*log10(Ns1c/Nsc_ideal))
Ds2c= 10^(log10(Dsc_ideal)-0.5*log10(Ns2c/Nsc_ideal))
Ds3c= 10^(log10(Dsc_ideal)-0.5*log10(Ns3c/Nsc_ideal))
Ds3E= 10^(log10(DsE_ideal)-1.0*log10(Ns3E/NsE_ideal))
Ds2E= 10^(log10(DsE_ideal)-1.0*log10(Ns2E/NsE_ideal))
Ds1E= 10^(log10(DsE_ideal)-1.0*log10(Ns1E/NsE_ideal))
display(' ')
D1c=Ds1c*(Q1c/ParallelStages_c1)^0.50/Ec1^0.25
D2c=Ds2c*(Q2c/ParallelStages_c2)^0.50/Ec2^0.25
D3c=Ds3c*(Q3c/ParallelStages_c3)^0.50/Ec3^0.25
```

```
D3E=Ds3E*(Q3E/ParallelStages_E3)^0.50/EE3^0.25
D2E=Ds2E*(Q2E/ParallelStages_E2)^0.50/EE2^0.25
D1E=Ds1E*(Q1E/ParallelStages_E1)^0.50/EE1^0.25
display(' ')
vtip1c=N1*D1c/2
vtip2c=N2*D2c/2
vtip3c=N3*D3c/2
vtip3E=N3*D3E/2
vtip2E=N2*D2E/2
vtip1E=N1*D1E/2
display(' ')
Re1c=vtip1c*D1c*p0/R/T0/Viscosity('MarsMix',T0)
Re2c=vtip2c*D2c*p2c_in/R/T2c_in/ Viscosity('MarsMix',T2c_in)
Re3c=vtip3c*D3c*p3c_in/R/T3c_in/ Viscosity('MarsMix',T3c_in)
Re3E=vtip3E*D3E*p3E_out/R/T3E_out/ Viscosity('MarsMix',T3E_out)
Re2E=vtip2E*D2E*p2E_out/R/T2E_out/ Viscosity('MarsMix',T2E_out)
Re1E=vtip1E*D1E*p1E_out/R/T1E_out/ Viscosity('MarsMix',T1E_out)
log10Re_val=[3 3.5 4 4.5 5 5.5 6 6.5 7 8];
f_eta_val=[0.2 0.34 0.5 0.65 0.8 0.915 0.97 0.99 1. 1.];
display(' ')
f_eta_1c=interp1(log10Re_val,f_eta_val,log10(Re1c),'linear','extrap' )
f_eta_2c=interp1(log10Re_val,f_eta_val,log10(Re2c),'linear','extrap' )
f_eta_3c=interp1(log10Re_val,f_eta_val,log10(Re3c),'linear','extrap' )
f_eta_3E=interp1(log10Re_val,f_eta_val,log10(Re3E),'linear','extrap' )
f_eta_2E=interp1(log10Re_val,f_eta_val,log10(Re2E),'linear','extrap' )
f_eta_1E=interp1(log10Re_val,f_eta_val,log10(Re1E),'linear','extrap' )

HorsePower =
       10000

ParallelStages_c1 =
     6
ParallelStages_c2 =
     1
ParallelStages_c3 =
     1
ParallelStages_E3 =
     1
ParallelStages_E2 =
     1
ParallelStages_E1 =
     6

Ec1 =
      127000
Ec2 =
      127000
Ec3 =
      127000
EE3 =
      250000
EE2 =
      295000
EE1 =
      295000
```

```
etac1 =
         0.838372361879645
etac2 =
         0.849129416377746
etac3 =
                      0.85
etaE3 =
         0.871143649531349
etaE2 =
         0.863408316502577
etaE1 =
         0.814743792449888

pic1 =
          11.2851768923859
pic2 =
          4.07517992348562
pic3 =
           2.9832761099289
piE3 =
          3.00698371395171
piE2 =
          4.79784979156976
piE1 =
          7.50026804574999

Wsp =
          262380.066940285
HeatAdded =
          748240.525292136
CycleEfficiency =
         0.350662732198104
mdot =
          28.4320378716033
ReactorPower =
          21274002.9521744

ShaftPower1c =
         -4306998.86336662
ShaftPower2c =
         -4252436.36605716
ShaftPower3c =
         -4248080.95258073
ShaftPower3E =
          6192097.30877051
ShaftPower2E =
          7241795.09627027
ShaftPower1E =
          6833623.77696374

Q1c =
          1374.03608265241
Q2c =
          223.186283300587
```

```
Q3c =
          75.9985899190462
Q3E =
          173.818130135216
Q2E =
          677.640320840184
Q1E =
          3913.39593664414

                   Pressures   Temperatures
ans =
                         850                         215
          9592.40035852798          394.108855322005
          39090.7573591095          546.891301403335
          116618.522548459          687.012148369157
          104956.670293613                      1290
          34904.3028755487          1116.85151782892
          7274.98867031615          907.513213906155
          969.963823418085          698.764323187268

NsE_ideal =
                        0.55
DsE_ideal =
                         3.3
Nsc_ideal =
                         0.7
Dsc_ideal =
                         3.7


N1 =
          291.614237899357
N2 =
          291.614237899357
N3 =
          503.302185542911

N1rpm =
          2784.71084625952
N2rpm =
          2784.71084625952
N3rpm =
          4806.18184188651

Ns1c =
         0.655963051075747
Ns2c =
         0.647574417580279
Ns3c =
         0.652196993986991
Ns3E =
         0.593500851812388
Ns2E =
          0.59971074764618
Ns1E =
```

```
         0.588360464331433

Ds1c =
          3.82217929409049
Ds2c =
          3.84685575791554
Ds3c =
          3.83319883673861
Ds3E =
          3.05812534970673
Ds2E =
          3.02645901732417
Ds1E =
          3.08484357809872

D1c =
          3.06396336906631
D2c =
          3.04430893282121
D3c =
            1.770162697523
D3E =
           1.8030917132241
D2E =
          3.38048422010396
D1E =
          3.38048422010396

vtip1c =
          446.747671410909
vtip2c =
          443.881914687431
vtip3c =
          445.463377214929
vtip3E =
                    453.75
vtip2E =
           492.89866478821
vtip1E =
           492.89866478821

Re1c =
          2558648.59248628
Re2c =
          8887696.15078288
Re3c =
          11656752.9698551
Re3E =
          3150642.55345375
Re2E =
          1899526.49334096
Re1E =
          396819.401004498

f_eta_1c =
```

```
         0.986320425740705
f_eta_2c =
           0.9989757839738
f_eta_3c =
     1
f_eta_3E =
          0.98993596537649
f_eta_2E =
         0.981145814207423
f_eta_1E =
         0.925845218693102
```

Results Extracted From MATLAB Output:

Summary of Engine Model #9 Turbocompressor Parameters At Full Design Point Operation

Output Power (US customary units)	**10,000 horsepower**					
Output Power (SI units)	**7.460 MW**					
Thermal Input Power (from Nuclear Reactor)	**21.274 MW**					
Thermal Conversion Cycle Energy Efficiency	**0.3507**					
Air Mass Flow Rate (kg/sec)	**28.432 kg/sec**					
	Low Pressure Stages Operating In Parallel		**Intermediate Pressure Stages**		**High Pressure Stages**	
Shaft Speed (N) (radians/sec)	**291.6 radians/sec**		**291.6 radians/sec**		**503.3 radians/sec**	
Shaft Speed (rpm)	**2,784.7 rpm**		**2,784.7 rpm**		**4,806.2 rpm**	
	Compressors (6)	**Turbines (6)**	**Compressor**	**Turbine**	**Compressor**	**Turbine**
D: Diameter of Rotor (m)	**3.064 m**	**3.380 m**	**3.044 m**	**3.380 m**	**1.770 m**	**1.803 m**
Rotor Tip Speed (m/s)	**446.7 m/s**	**492.9 m/s**	**443.9 m/s**	**492.9 m/s**	**445.5 m/s**	**453.8 m/s**
E: Reversible Fluid Work Per Unit Mass (kJ/kg)	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**295 kJ/kg**	**127 kJ/kg**	**250 kJ/kg**
Ds: Specific Diameter	**3.82**	**3.08**	**3.85**	**3.03**	**3.84**	**3.06**
Ns: Specific Speed	**0.656**	**0.588**	**0.648**	**0.600**	**0.652**	**0.594**
Est. Efficiency From Balje Diagram	**0.85**	**0.88**	**0.85**	**0.88**	**0.85**	**0.88**
Machine Reynolds Number	**$2.56*10^6$**	**$3.97*10^5$**	**$8.89*10^6$**	**$1.90*10^6$**	**$1.17*10^7$**	**$3.15*10^6$**
Reynolds Number Efficiency Factor	**0.986**	**0.926**	**0.999**	**0.981**	**1.000**	**0.990**
Assumed Efficiency (from Balje*Reynolds)	**0.838**	**0.815**	**0.849**	**0.863**	**0.850**	**0.871**
Intake Air Temperature (Kelvins)	**215 Kelvins**	**907.5Kelvins**	**394.1 Kelvins**	**1116.9 Kelvins**	**546.9 Kelvins**	**1290 Kelvins**
Intake Air Pressure (Pascals)	**850 Pa**	**7,275 Pa**	**9,592 Pa**	**34,904 Pa**	**39,091 Pa**	**104,957 Pa**
Pressure Ratio (Calculated)	**11.285**	**7.460**	**4.072**	**4.791**	**2.981**	**3.007**
Discharge Air Temperature (Kelvins)	**394.1 Kelvins**	**698.8 Kelvins**	**546.9 Kelvins**	**907.5Kelvins**	**687.0 Kelvins**	**1116.9 Kelvins**
Discharge Air Pressure (Pascals)	**9,592 Pa**	**970 Pa**	**39,091 Pa**	**7,275 Pa**	**116,619 Pa**	**34,904 Pa**
Q:Volumetric Flow Rate Total Low Pressure Sides	**1,374.04 m^3/s**	**3913.40 m^3/s**	**223.19 m^3/s**	**677.64 m^3/s**	**76.00 m^3/s**	**173.82 m^3/s**
Shaft Powers (kW) [Total=7.460MW=10,000hp]	**-4.307 MW**	**+6.834 MW**	**-4.252 MW**	**+7.242 MW**	**-4.248 MW**	**+6.192 MW**

APPENDIX F
F MATLAB: HEAT EXCHANGER DESIGN CALCS

This MATLAB notebook is devoted to solving 1-D duct-flow problems in heat exchanger design analyses. The differential equations solved herein follow the text, Compressible Fluid Flow by Michel A. Saad[1985], and in particular, this text's section 6.10 FLOW IN A FRICTIONAL CONSTANT AREA DUCT WITH HEAT TRANSFER. This text's section 6.10 starts with a series of transformations eliminating pressure, temperature, and speed variables for the flowing gas and replacing them instead with M, the local "Mach number" which is the ratio of the local gas speed to the local sound speed, and To, the "stagnation temperature" (which in some other texts is called the "total temperature"), which is the temperature the flowing gas would reach if it were adiabatically brought to rest. For instance, it invokes the following definition of stagnation temperature:

$$T_o \equiv T\left(1+\frac{\kappa-1}{2}M^2\right) \tag{F-1}$$

where κ is the gas's ratio of specivic heats, Cp/Cv. These two quantities, M, and To, vary as a function of position in the duct, where the position is expressed as the distance from the duct entrnce, denoted by the symbol, x. Thus, the problems solved herein all involve determining the functions, M(x) and To(x), which are found by solving Ordinary Differential Equations (ODEs). After determining them, inverse transformations are applied to calculate the pressure profile, p(x), the speed profile v(x), the (true) temperature profile, T(x), and the density profile, ρ(x).

The fundamental ODE derived in Saads' text (Saad's Eq, 6.99) for the evolution of a gas flow's Mach number in a constant area duct is stated in differential form as follows:

$$\frac{dM^2}{M^2} = \frac{\left(1+\kappa M^2\right)\left(1+\frac{\kappa-1}{2}M^2\right)}{1-M^2}\frac{dT_o}{T_o} + \frac{\left(\kappa M^2\right)\left(1+\frac{\kappa-1}{2}M^2\right)}{1-M^2}\frac{4f}{D}dx \tag{F-2}$$

Here, , D is the hydraulic diameter of the constant-area duct which is defined as 4 times the area of the duct's cross-section (perpendicular to the flow) divided by the perimeter of the same cross-section. The quantity, *4f*, is the friction factor for flow in pipes as given by Moody's diagram. Note that many texts on this subject refer to this same friction factor as simply *f* while other texts follow the *4f* convention used here, so it is necessary to use caution when interpreting multiple texts. For turbulent flow in smooth ducts, the friction factor can be approximated by solving an iterative equation attributed to Karman-Nikuradse:

$$\frac{1}{\sqrt{4f}} + 0.8 + 2\log_{10}\left(\mathrm{Re}_D \sqrt{4f}\right) = 0 \tag{F-3}$$

where Re_D is the local Reynolds number of the flow, given by

$$\mathrm{Re}_D \equiv \frac{\rho v D}{\mu} = \frac{4\dot{m}}{P\mu} = \frac{4\dot{m}}{\pi D \mu} \tag{F-4}$$

where ρ represents mass density of the flowing gas, μ represents gas viscosity,, v represents gas speed, P represents the perimenter of the duct's cross section and mdot is the constant total mass flowrate through any cross section in the duct. Thus, the Reynolds number only varies along the duct because of changes in gas viscosity, which in turn is only affected by gas temperature.

The second RHS term in Saad's Eq 6-99 represents the effect of friction with the walls. The first term represents the effect of increasing the stagnation temperature, which in this model results from adding heat through the duct walls. The heat added from the wall between two differentially separated stations is transferred to the entrained fluid, as follows:

$$\delta\dot{Q} = (T_w - T_{aw}) h_c P dx = \rho A v c_p dT_o \tag{F-5}$$

where T_w is the wall temperature and T_{aw} is the adiabatic gas temperature adjacent to the wall, given by

$$T_{aw} = r_f T_o + (1 - r_f) T \tag{F-6}$$

where r_f is an empirically defined "recovery factor" which Saad states is typically between 0.83 and 0.91 for flat plates, and where h_c is the heat transfer coefficient. Combineing these equations results in the following ODE:

$$\frac{dT_o}{T_w - r_f T_o + (1 - r_f) T} = \frac{4h_c}{\rho c_p v D} dx \tag{F-7}$$

Of course, calculations are simplified if the recovery factor is approximated as unity.

There are various correlations to predict the heat transfer coefficient which have been used for gases. The one developed by Saad and used here is called Reynolds Analogy:

$$h_c = \frac{f}{2} \rho c_p v \tag{F-8}$$

Reynolds Analogy is theoretically based on the fact that on a molecular level, the diffusion mechanism responsible *in a gas* for viscosity is the same diffusion mechanism that is responsible for heat transfer. It is a model frequently assumed for gaseous fluids in many texts, although more precise heat transfer coefficient models for specific gases are also in frequent use. Substituting Reynolds Analogy into the previous result yields:

$$\frac{2dT_o}{T_w - r_f T_o + (1 - r_f)T} = \frac{4f}{D}dx \tag{F-9}$$

For the special case where wall temperature is spatially constant, where the recovery factor is approximated as unity, and where the friction factor is also taken as a constant (at least approximately), this can be solved in closed form. For other cases, and in particular the present case, this ODE is coupled to other ODEs and also to various algebraic formulae interrelating the variables.

The models developed in Saad's text are specifically for a perfect gas having specific heats which are constant, i.e., which are independent of temperature. This model is completely accurate for some gases, e.g., helium. However, some other gases such as for instance CO_2 have specific heats which vary widely with temperature, so for those gases this model would be inaccurate. The approach taken here is to approximate the gas's variable specific heat ratio over a temperature range of interest as being a constant value while using Saad's equations.

It is also necessary to relate Saad's duct flow model to the change in temperature of the flowing lithium reactor coolant.

$$d\dot{Q} = -\dot{m}_{Li} c_p dT_{Li} \tag{F-10}$$

M-files exist for lithium's specific heat and enthalpy, so there are alternative ways to state this. However, in either case there must also be a modeled temperature drop in the film from Li to solid, and also another temperature drop within the wall. We will assume the lithium flows in its own nominally circular tube with a different diameter than the tube carrying the gas:

Thus,

$$d\dot{Q} = h_C^{Li-wLi}(T_{Li} - T_{wLi})P_{Li}dx = h_C^{Li-w}(T_{Li} - T_{wLi})\pi D_{Li}dx \tag{F-11}$$

If the wall were merely a thin solid metal layer separating the liquid lithium from the gas, we would have the following:

$$d\dot{Q} = k_{wall}\left(\frac{T_{wLi} - T_w}{\tau_{wall}}\right)P_{Li}dx \tag{F-12}$$

With the actual thermal connection between liquid lithium and gas tubes, a more complicated relation may be more appropriate. However, we will use this model and increase the effective wall thickness to approximately model the actual situation.

For heat transfer between the liquid lithium and its solid wall we will use the Lyon-Martinelli correlation [Duderstadt and Hamilton 1976, 486]:

$$Nu = 6.3 + 0.030(\mathrm{Re}\,\mathrm{Pr})^{0.8} \tag{F-13}$$

where

$$\mathrm{Pr} \equiv \frac{\mu c_p}{k} \tag{F-14}$$

and where

$$\mathrm{Re} \equiv \frac{\rho v D_{Li}}{\mu} = \frac{4\dot{m}_{Li}}{\pi D_{Li}\mu} \tag{F-15}$$

and where

$$h_c^{Li-wLi} \equiv \frac{k}{D_{Li}} Nu \tag{F-16}$$

COMPUTATION

At each station along the heat exchanger, we start the calculation for that station using assumed values of M, T_{Li} and T_o.
(1) Get the viscosity of liquid lithium, based on T_{Li} .
(2) Calculate Reynolds number of the lithium flow.
(3) Also get the thermal conductivity and specific heat of liquid lithium, based on T_{Li}
(4) Calculate the Prandtl Number of liq. lithium.
(5) Evaluate Lyon-Martinelli correlation for Nu
(6) Calculate local heat transfer coefficient from lithium to its wall.
(At this point we have the heat flow per unit temperature difference per unit length, lithium to its wall)
(7) Calculate true temperature of gas from stagnation temperature and Mach number
(8) Calculate viscosity of gas from its true temperature
(9) Calculate Reynolds number of gas flow
(10) Calculate friction factor, f
(11) Calculate heat flow per unit temperature difference per unit length for gas flow to wall
(12) Using constant "wall" thermal conductance, solve for total heat flow and also for the wall temperature for the gas flow.

A single cell implementation could be as in the following MATLAB code.
Input parameters are mdotLi, mdot, kappa,cp, D,DLi,tauwall, kwall

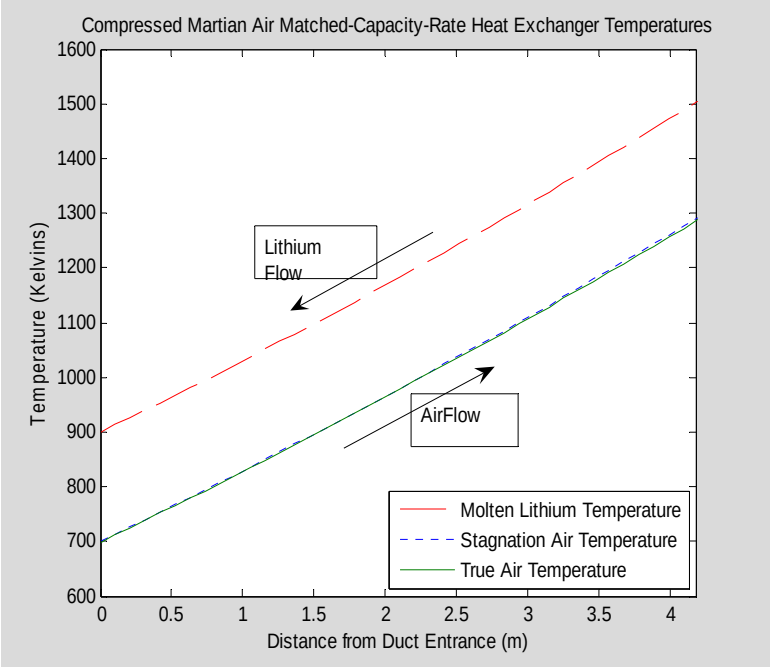
Input variables are To, TLi, M2

```
R=GasConstant('MarsMix')
cp=(Enthalpy('MarsMix',1290)-Enthalpy('MarsMix',700))/(1290-700)
kappa=cp/(cp-R)
D=0.1;DLi=0.01;
tauwall=0.01;kwall=300.
mdotLi=1;mdot=1;


To=700;
TLi=900;
M2=0.01;

muLi=Viscosity('Li',TLi);
ReLi=4/pi*mdotLi/DLi/muLi;
kLi=ThermalConductivity('Li',TLi);
cpLi=SpecificHeat('Li',TLi);
PrLi=muLi*cpLi/kLi;
NuLi=6.3+0.030*(ReLi*PrLi)^0.8
hcLi=NuLi*kLi/DLi;
T=To/(1+M2*(kappa-1)/2)
mu=Viscosity('MarsMix',T);
Re=4/pi*mdot/D/mu;
sqrt4f=1;
for i=1:10
sqrt4f=1/(0.8+log10(Re*sqrt4f))^2;
end
f=sqrt4f^2/4
Rgas=1/(4*mdot/D*f/2*cp)
Rwall=tauwall/kwall
RLi=1/(hcLi*pi*DLi)
dQdotdx=(TLi-To)/(Rgas+Rwall+RLi)
Tw=Rgas*dQdotdx+To
dTLidx=-dQdotdx/mdotLi/cpLi;
dTodx=2*f*(Tw-To)/D;
dM2dx=M2*dTodx/To*(1+(kappa-1)/2*M2)/(1-
M2)*(1+kappa*M2+kappa*M2*2*To/(Tw-To))

R =
  191.0604
cp =
  1.2059e+003
kappa =
    1.1883
kwall =
   300
NuLi =
   50.9886
T =
  699.3417
f =
  3.9397e-004
Rgas =
```

```
    0.1052
Rwall =
  3.3333e-005
RLi =
  1.0917e-004
dQdotdx =
  1.8978e+003
Tw =
  899.7296
dM2dx =
  2.4894e-005
```

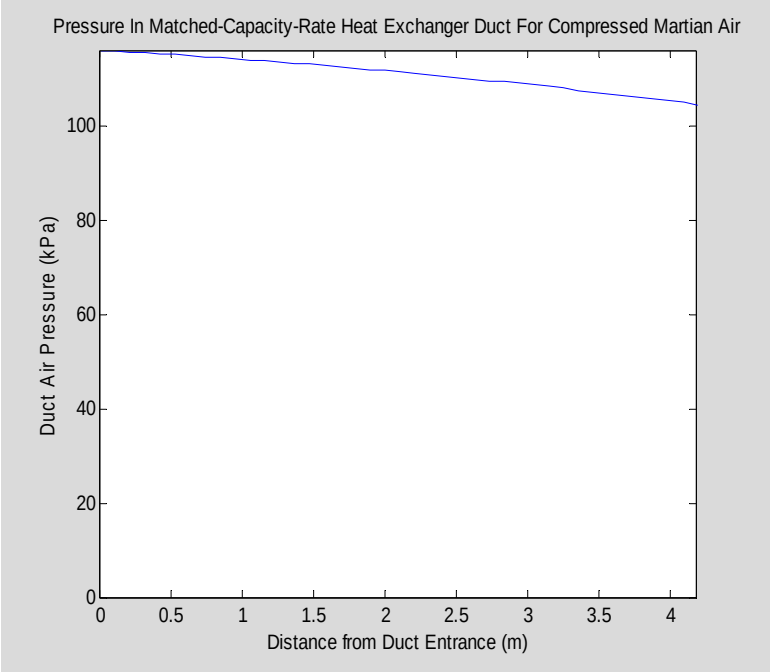
```
D=0.005;DLi=0.002;tauwall=0.01;kwall=300;mdotratio=3.4465;
p0=116000;
To0=700;
TLi0=900;
M0=0.125;
M20=M0^2;
xend=4.2;
parameters(1)=D;
parameters(2)=DLi;
parameters(3)=tauwall;
parameters(4)=kwall;
parameters(5)=p0;
parameters(6)=mdotratio;
parameters(7)=To0;
parameters(8)=TLi0;
parameters(9)=M20;
parameters(10)=xend;
[T,pratio,v,X,Y]=Hxode(parameters);
plot(X,Y(:,3),'--r',X,Y(:,2),':',X,T)
xlim([0 4.18])
title('Compressed Martian Air Matched-Capacity-Rate Heat Exchanger
Temperatures')
xlabel('Distance from Duct Entrance (m)')
ylabel('Temperature (Kelvins)')
legend('Molten Lithium Temperature','Stagnation Air Temperature','True
Air Temperature','Location','SouthEast')

R =
                191.06037
cp =
          1205.88377260899
kappa =
          1.18826957430111
D =
                    0.005
DLi =
                    0.002
tauwall =
                     0.01
```

```
kwall =
   300
p0 =
      116000
mdotratio =
                    3.4465
To0 =
   700
TLi0 =
   900
M20 =
                  0.015625
xend =
                       4.2
T0 =
          698.971912933622
rhogas0 =
         0.868615650508119
v0 =
          49.7946485111603
mdot =
      0.000849259104211637
Re0 =
          7087.40153039912
mdotLi =
      0.000246412042423223
vLi =
         0.165500896593287
```

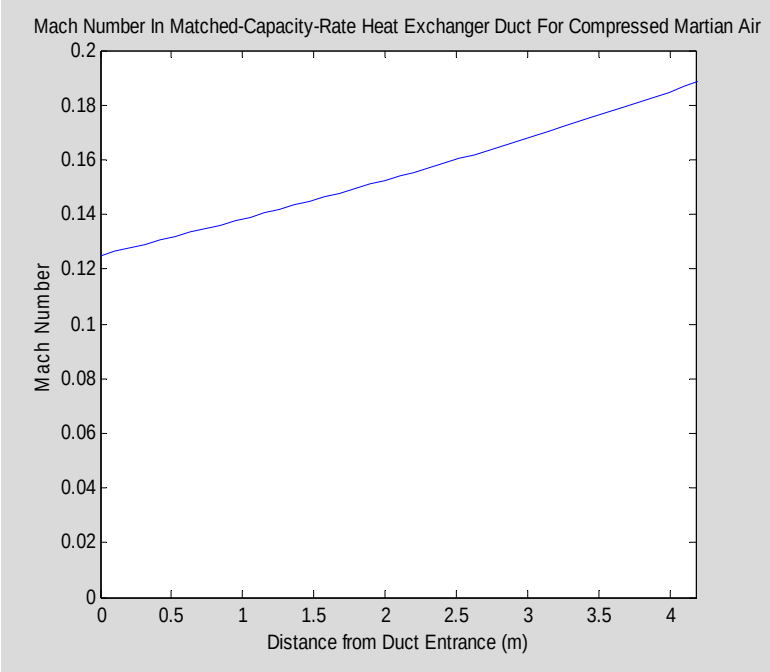



```
plot(X,116*pratio)
xlim([0 4.18])
```

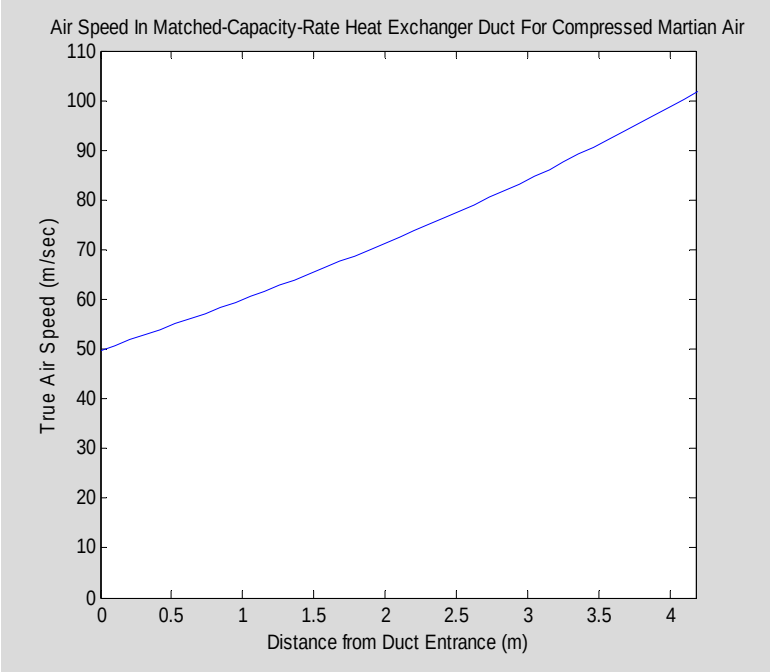
```
ylim([0 116])
xlabel('Distance from Duct Entrance (m)')
ylabel('Duct Air Pressure (kPa)')
  title('Pressure In Matched-Capacity-Rate Heat Exchanger Duct For
Compressed Martian Air')
```




```
plot(X,sqrt(Y(:,1)))
xlim([0 4.18])
ylim([0 0.2])
xlabel('Distance from Duct Entrance (m)')
ylabel('Mach Number')
title('Mach Number In Matched-Capacity-Rate Heat Exchanger Duct For
Compressed Martian Air')
```




```
plot(X,v)
xlim([0 4.18])
ylim([0 110])
xlabel('Distance from Duct Entrance (m)')
ylabel('True Air Speed (m/sec)')
title('Air Speed In Matched-Capacity-Rate Heat Exchanger Duct For
Compressed Martian Air')
```



APPENDIX G
G CANDIDATE MODERATOR/SHIELDING MATERIALS

The following initial list of candidate materials along with apparent useful descriptive information for each was prepared by consulting various reference publications. The question this list addresses is which materials have nuclear engineers found in the past to be useful for projects with similarities to this nuclear engine design, and what special properties must be considered in their application. The purpose of compiling this initial list was to avoid wasting time investigating the use of nonoptimal materials.

G.1 Beryllium-9

Beryllium has only one stable isotope, Be-9. Figure G-1 shows the elastic cross section and the total cross section of Be-9. The two curves overlay each other below about 1 MeV and deviate slightly for higher energies. Beryllium is a metallic element that has been used in some high temperature reactor designs as a neutron moderator. Its moderating power is high compared with most other elements except for hydrogen. Also, its capture cross section for thermal neutrons is small, only 0.008 barns [KAPL CoN]. Its density at 25 C is 1.85 g/cm^3, its linear expansion coefficient is 11.3*10^{-6} /Kelvin, its melting and boiling points are 1560 and 2744 Kelvins (i.e., 1287 and 2471 C), while its solid phase thermal conductivity is 2.00 W cm^{-1} Kelvin^{-1} [CRC].

G.2 Boron-10

Natural boron is 20 % B^{10} and 80% B^{11}. The main interest in using boron is in the neutron absorption properties of the B^{10} . Figure G-2 shows the elastic and total cross section of B-10. Their deviation below 1 MeV represents the neutron absorption reaction, $B^{10}(n,\alpha\gamma)Li^7$ which produces a gamma ray photon with 0.5 MeV energy. Boron is a Group 13 element, with chemical properties intermediate between metals and non-metals. Boron has a rich chemistry, forming hydride molecular chains and also forming chlorides, fluorides, carbides, oxides, sulfides, and nitrides. However, in its solid crystaline form it is essentially inert, resisting attack even by boiling HCl or HF acids. The density of solid boron with the natural isotopic mix is 2460 kg/m^3, so the density of solid B^{10} can be calculated as 2278 kg/m^3, i.e.2,278 g/cm^3. The melting point of boron is 2349 Kelvins (i.e., 2076 C), and it boils at 4200 Kelvins.

G.3 Boron Carbide

Boron carbide is the boron-bearing solid material commonly used for the neutron absorber, boron, in control rods, shutdown pellets, safety rods, and neutron shielding. Its density using natural boron and carbon is 2.52 g/cm^3 and it melts at 2718 Kelvins (i.e., 2445 C.).

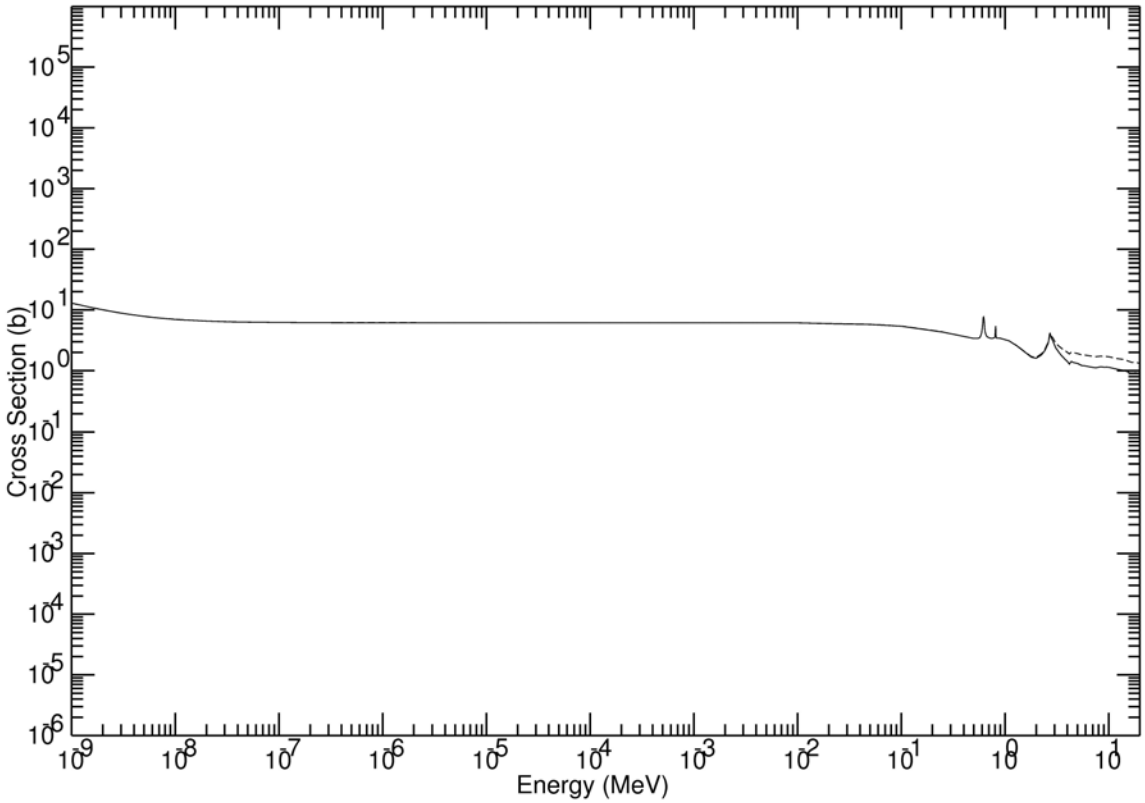


Figure G-1: Scattering & Total Neutron Cross Sections Of Be-9
(From http://atom.kaeri.re.kr/ using ENDF-VI Library)

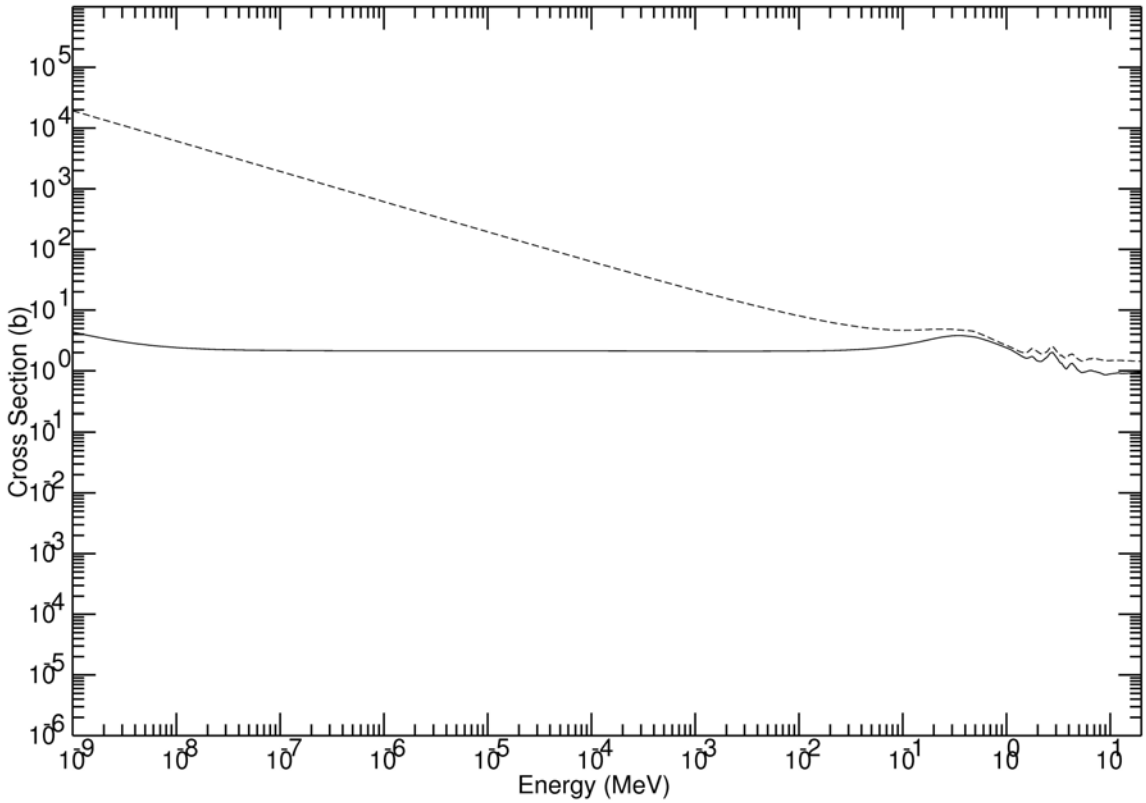


Figure G-2: Scattering and Total Neutron Cross Sections Of B-10
(From http://atom.kaeri.re.kr/ using pointwise ENDF-VI Library)

G.4 Carbon

Figure G-3 shows the elastic and total neutron cross sections of natural carbon. Graphite is used extensively in some stationary reactor designs because it is a fairly good neutron moderator and reflector material, although it not a good gamma ray attenuator. Pure carbon exists in nature as either diamond or graphite. Diamond is not presently available in large sizes or in arbitrary shapes. Carbon's graphite form has great strength and stability from room temperature to very high temperatures. It resists cracking, warping, shrinking, or distortion even when temperatures exceed 2773 Kelvins (i.e., 2500 C). Its high strength allows it to be used as a structural material. It can also be processed into a variety of forms from felts and foils to composites and precisely machined components. Graphite is resistant to thermal shock. It is anisotropic, with high thermal conductivity in one direction and low thermal conductivity in a perpendicular direction, so it can act as either a thermal conductor or an insulator depending on the application. Densities of graphites vary but are typically 2.2 g/cm^3. Graphite is electrically conductive and naturally lubricating. Graphite is chemically almost inert in most environments.

G.5 Hafnium

Hafnium is a component of some refractory alloys. Naturally occurring hafnium isotopes have the following abundances: 0.18% Hf-174, 5.2 % Hf-176, 18.6% Hf-177, 27.1% Hf-178, 13.7 % Hf-179, 35.2% Hf-180 [KAPL CoN]. Figure G-4 shows the elastic and total neutron cross sections of hafnium's natural blend of isotopes. Hafnium melts at 2506 Kelvins (i.e., 2233 C). Its density is 13.3 g/cm^3 at 25 C, linear expansion coefficient is $5.9*10^{-6}$ $Kelvin^{-1}$, and thermal conductivity is $0.230 Wcm^{-1}Kelvin^{-1}$ [CRC].

G.6 Hydrogen

Figure G-5 shows radiative capture, elastic scattering and total neutron cross sections of hydrogen, the scattering and total cross section curves overlay each other. Hydrogen has a high scattering cross section compared to other elements, 38 barns for thermal neutrons and more than 20 barns for neutron energies up to 20 keV. For higher energies hydrogen's scattering cross section drops off with an approximate $E^{-0.5}$ energy dependence, falling below 1 barn at 10 MeV.

Radiative neutron capture in hydrogen has a considerably smaller cross section than elastic scattering and has an approximate $E^{-0.5}$ energy dependence. A 2.2 MeV gamma ray photon is released when hydrogen absorbs a neutron.

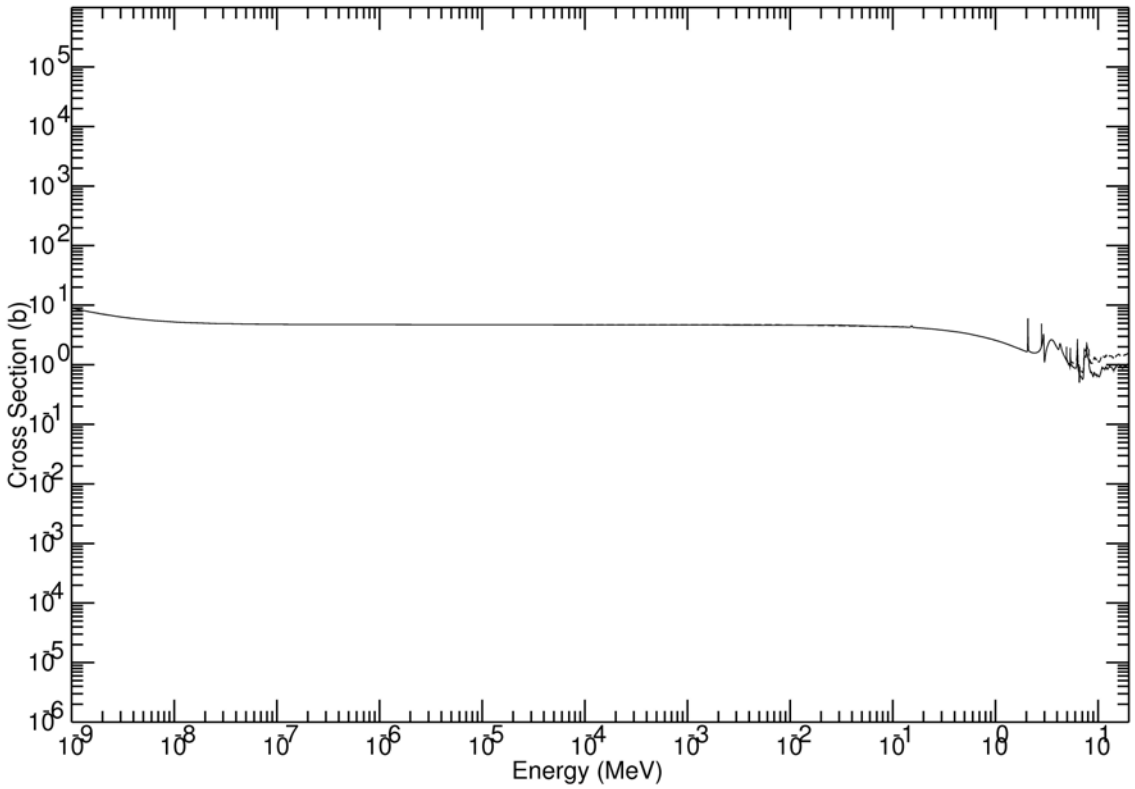


Figure G-3: Carbon Scattering &Total Neutron Cross Sections
(From http://atom.kaeri.re.kr/ using ENDF-VI Library)

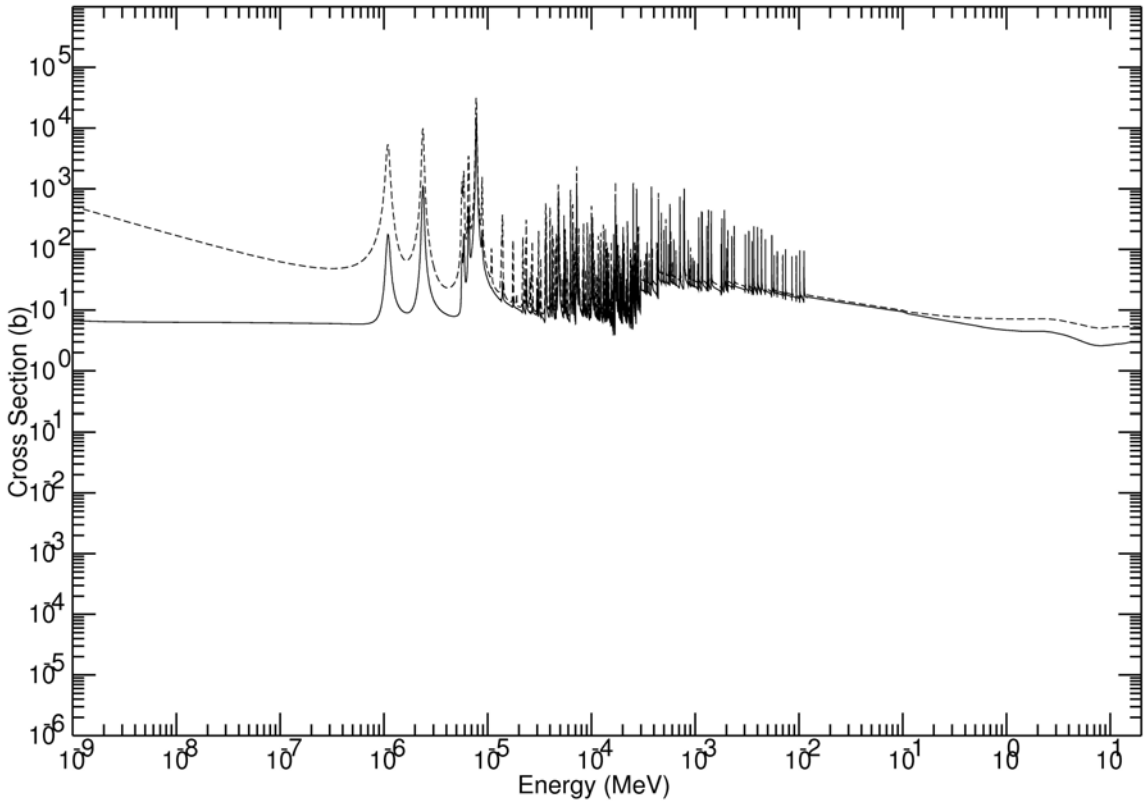


Figure G-4: Hafnium Scattering&Total Neutron Cross Sections
(From http://atom.kaeri.re.kr/ using ENDF-VI Library)

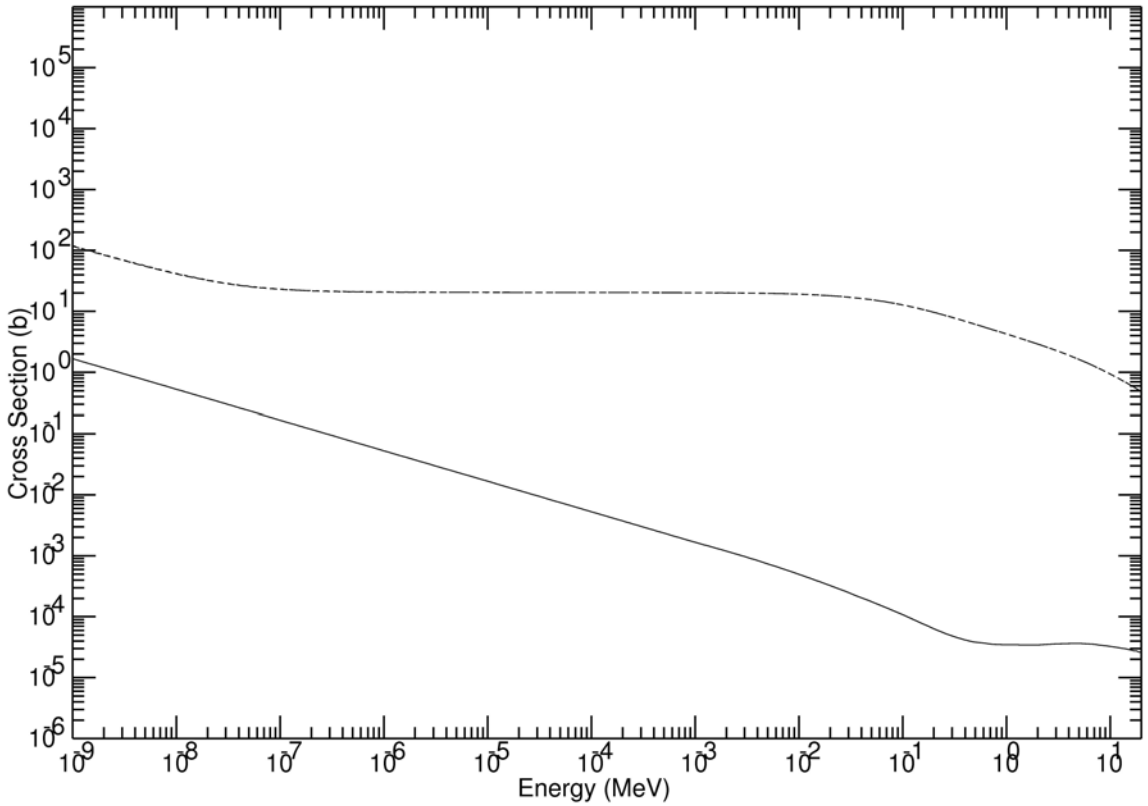


Figure G-5: Hydrogen Capture, Scattering & Total Cross Sections
(From http://atom.kaeri.re.kr/ using ENDF-VI Library)

G.7 Iron

Iron is not listed in Radiation Shielding as a material typically used in mobile reactors, although it is ubiquitous in stationary reactors for many reasons, e.g., structure. Although it is mechanically useful at room temperature and slightly elevated temperatures it rapidly loses strength at higher temperatures. By 823 Kelvins (i.e.550 C) it has lost half of its 25 C yield and ultimate tensile strengths, and its remaining strength drops precipitously as temperature is increased further. Iron melts at 1811 Kelvins (i.e., 1538 C), has a density at 25 C of 7.87 g/cm^3 and has a linear expansion coefficient of 11.8*10^{-6} Kelvin^{-1} [CRC]. Iron is also useful in the generation of magnetic fields due to its ferromagnetic properties which confer asymptotic low-temperature magnetization strength of 1.75 Tesla. In the present nuclear engine design iron's magnetic properties could help with coolant pumping. Iron's saturation magnetization decreases with increasing temperature, disappearing altogether at the Curie temperature which for pure iron occurs at 1043 Kelvins (i.e., 770 C) and which occurs at nearby temperatures for steel alloys.

Iron's neutron and gamma shielding properties are good but not exceptional. Inelastic scattering's "removal" cross section combined with a hydrogenous material is not as high as tungsten's on a macroscopic basis but is double tungsten's on a mass basis.

Natural iron isotope abundances are 5.8% Fe-54, 91.8% Fe-56, 2.1% Fe-57, and 0.3% Fe-58. Of these, Fe-58 is easily activated by thermal neutrons to Fe-59 which intensely emits 1.1 MeV and 1.3 MeV gamma rays with a 44.6 day half-life [CoN by KAPL]. Figure G-6 shows the elastic and total neutron cross sections of iron isotopes Fe-54, Fe56, Fe57, and Fe-58.

G.8 Iron Oxide/Silica

This material is listed here because it is available almost everywhere on the martian surface in large quantities as the fine dust, iron oxide (Fe_3O_4) mixed with silica (SiO_2). This material may comprise most of the airborne dust in martian dust storms. It could be incorporated into radiation shielding without paying the huge cost of transport from Earth, since its only cost would be for "vacuuming" it from the martian surface and depositing it in containers as part of the engine's shielding. It is not expected to be particularly efficient as a shielding material but on the other hand it would be almost free.

G.9 Lead

Figure G-7 shows the elastic and total neutron cross sections for lead isotopes, Pb-206, Pb-207, and Pb-208. Natural lead isotope abundances are 1.42% Pb-204, an isotope with an estimated half-life of 1.4*10^{17} years, 24.1% stable Pb-206, 22.1% stable Pb-207, and 52. % stable Pb-208. Lead melts at 600 Kelvins (i.e.,327 C), boils at 2026 Kelvins (1749C) at 1 atmosphere, has a density of 11.3 g/cm^3, and a linear expansion coefficient of 28.*10^{-6} Kelvin^{-1}. Lead is an excellent absorber of gamma-rays.

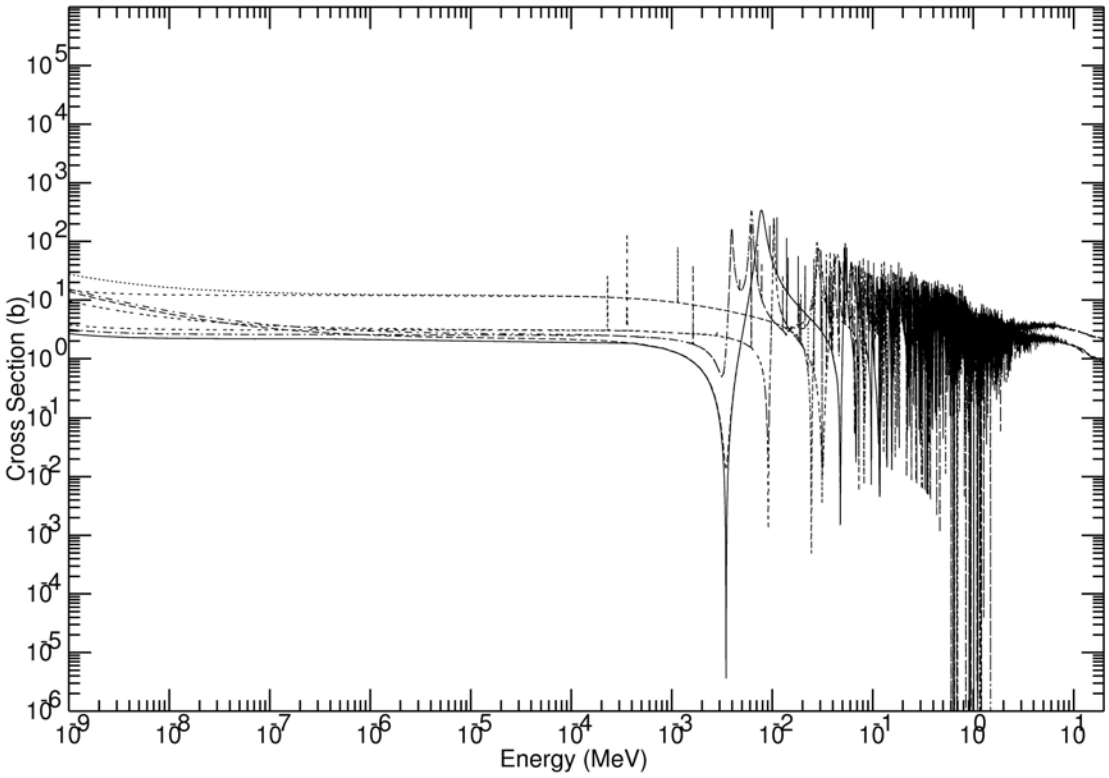


Figure G-6: Scattering&Total CrossSections of Fe-54, 56, 57,58. (From http://atom.kaeri.re.kr/ using ENDF-VI Library)

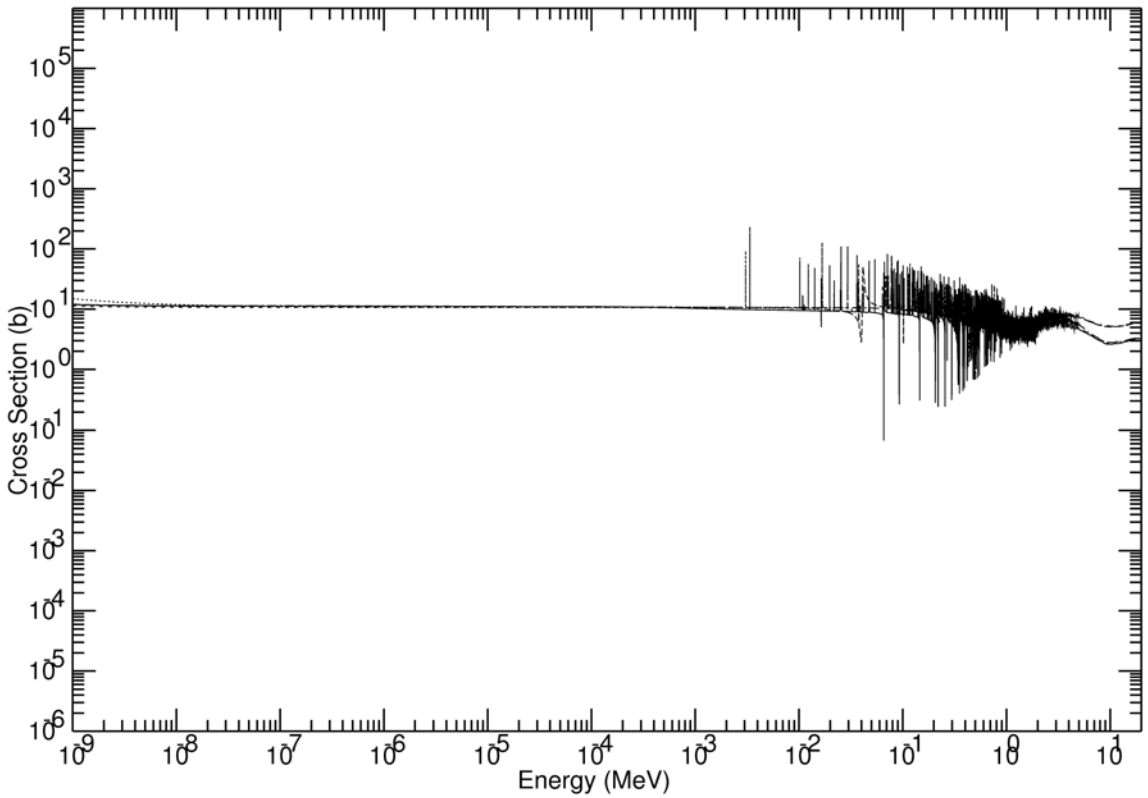


Figure G-7: Scattering and Total Neutron Cross Sections of Pb-206,207, & 208. (From http://atom.kaeri.re.kr/ using pointwise ENDF-VI Library)

G.10 Lithium-6

The lithium-6 isotope comprises 7.5% of naturally occuring lithium. Figure G-8 shows the elastic cross section and the total cross section of Li-6. Their large deviation below 1 MeV represents the neutron absorption reaction, $Li^6(n,\alpha t)$, which does not produce any gamma ray photons.

G.11 Lithium-7

The lithium-7 isotope comprises 92.5% of natural lithium. Figure G-9 shows the elastic and total cross sections of Li-7.

G.12 Lithium Hydride

Lithium hydride, LiH, is a material with a high hydrogen density which can be confined at low pressures to ssignificantly higher temperatures than other hydrogen-bearing materials such as water or polyethylene plastic. At 25 C with the natural isotopic mix LiH solid density of 0.775 g/cm^3, its hydrogen atom density is $5.87*10^{22}$ $atoms/cm^3$. This can be compared with water's hydrogen density of $6.7*10^{22}$ $atoms/cm^3$ at water's nominal density of 1.0 g/cm^3.

With isotopic tailoring LiH has been chosen in high temperature nuclear reactor designs as a neutron moderator and as a component of radiation shielding. LiH, incorporating the natural blend of 7.5% Li^6 and 92.5% Li^7 isotopes, melts and freezes at 961 Kelvins (i.e., 688 C). At Earth's sea level atmospheric pressure LiH decomposes at 1123 Kelvins (i.e., 850 C), but its decomposition can be entirely prevented up to considerably higher temperature by confining the LiH in contact with pressurized hydrogen gas. In a vacuum LiH decomposes at lower temperatures.

Some data on the density vs temperature of LiH using natural lithium was found in Table 6.10 on page 219 of Metal Hydrides by [Mueller, Blackledge, and Libowitz 1968, 210], and has been summarized in Table G-1 and Figure G-10. Note that the 1223 Kelvins (950 C) data point reflects measurements made under a pressurized hydrogen gas blanket in order to avoid LiH decomposition.

LiH is even more chemically reactive than molten lithium. It readily reacts with oxides, chlorides, sulfides and also reacts with ceramics at high temperature, but it is chemically compatible with refractory metals at high temperature [TC-1000]. Pure LiH is not a good electrical conductor so if a reactor design needed to circulate pure liquid LiH then electromagnetic pumping would not be efficient; some form of mechanical pump would be needed. In its solid form, LiH has been used in various applications as a neutron absorber.

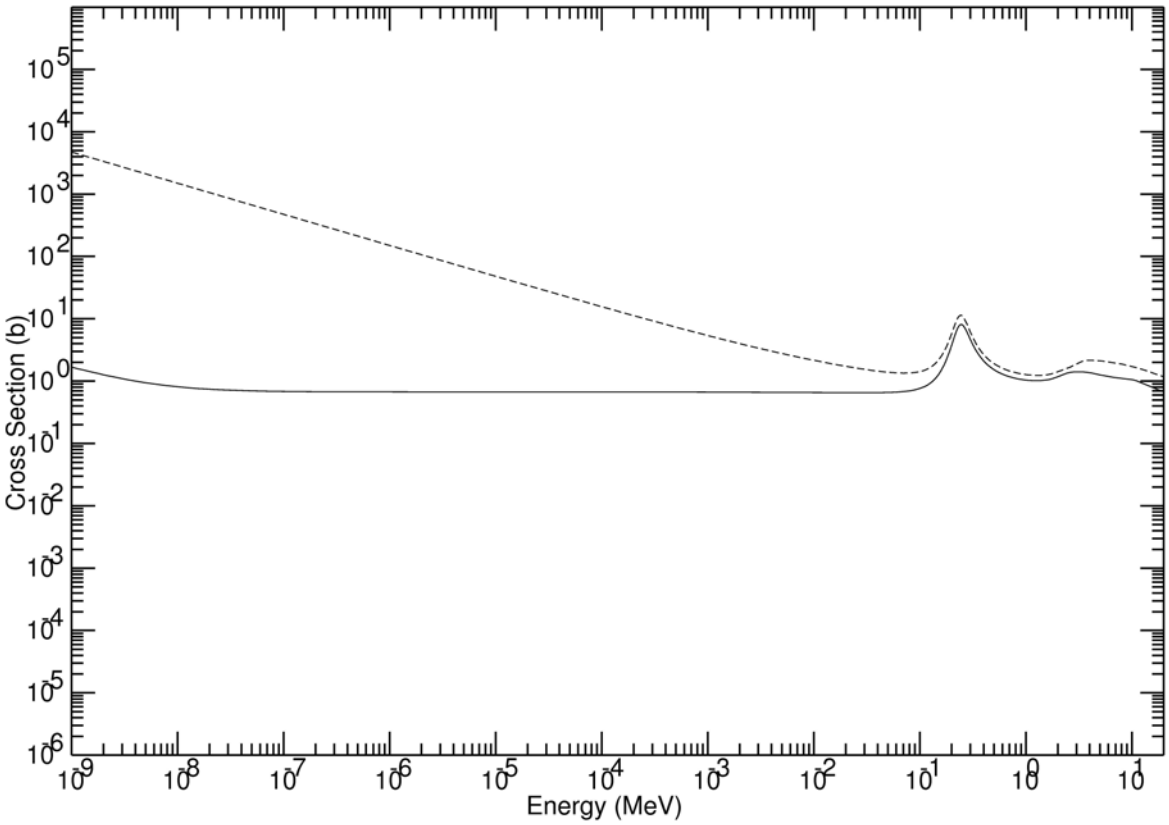


Figure G-8: Scattering and Total Neutron Cross Sections Of Li-6
(From http://atom.kaeri.re.kr/ using pointwise ENDF-VI Library)

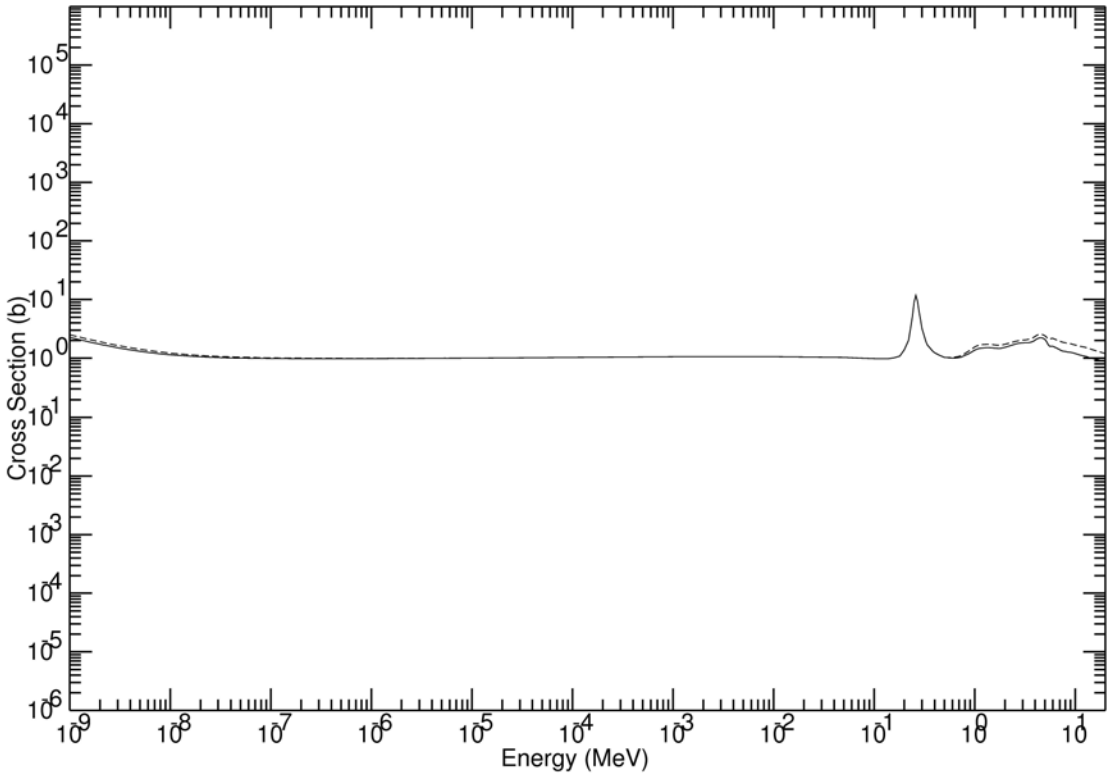


Figure G-9: Scattering & Total Cross Sections Of Lithium-7
(From http://atom.kaeri.re.kr/ using ENDF-VI Library)

Table G-1: Natural Lithium Hydride
Density Vs. Temperature

Temperature (C)	Lattice Parameter, a0 (Angstrom)	Density (g/cm^3)
25	4.083	0.775
200	4.116	0.757
400	4.159	0.733
525	4.188	0.718
600	4.21	0.705
688 (solid)	4.21	0.69
700 (liquied)	--	0.550
950 (liquid)	--	0.492

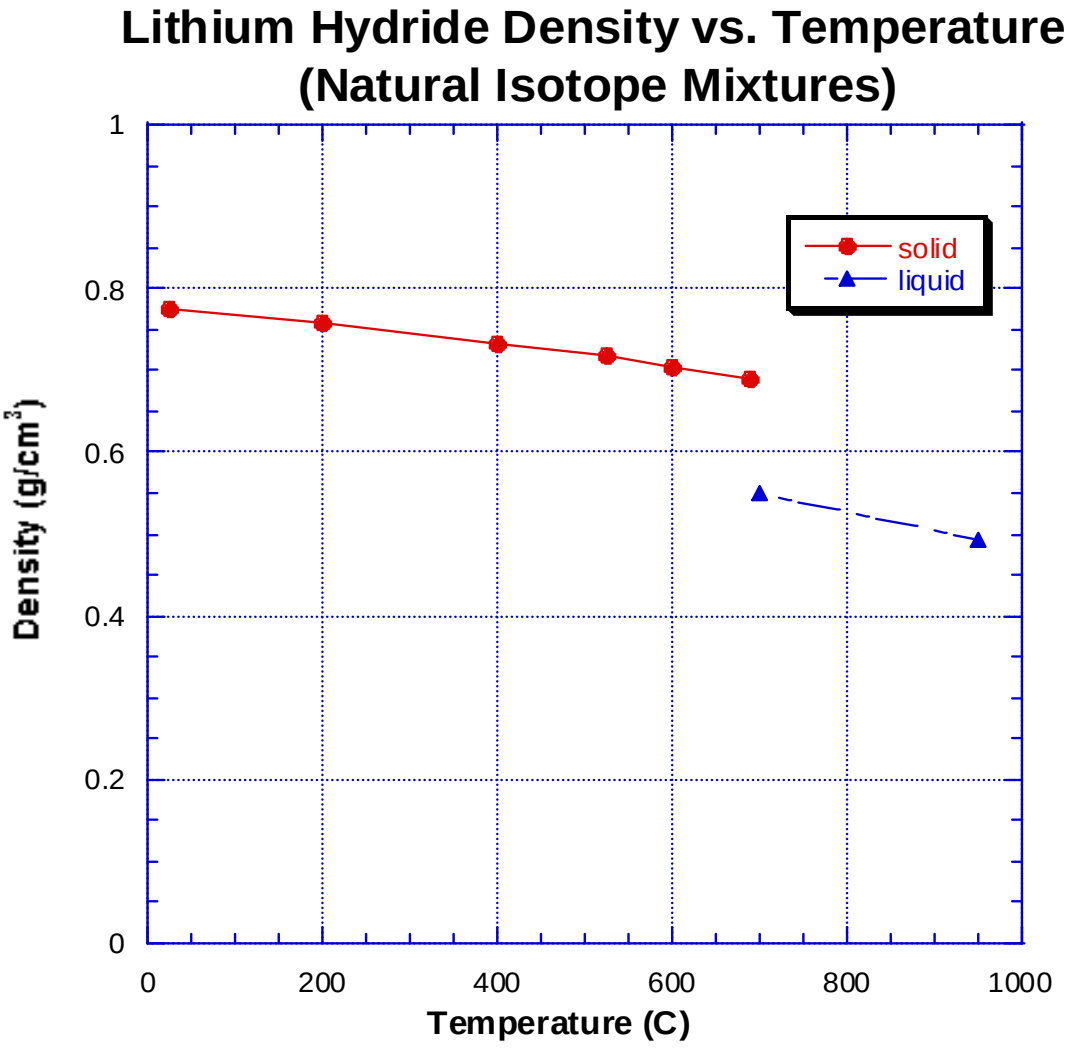


Figure G-10: Natural Lithium Hydride Density Vs. Temperature

G.13 Lithium-6 Hydride

This is an isotopically purified version of lithium hydride (LiH). The atom densities in lithium-6 hydride are identical to natural lithium hydride, therefore the mass density of lithium-6 hydride is about 88% of natural lithium hydride's mass density as plotted in Figure G-10. The Li^6 isotope is useful for shielding because it absorbs neutrons without producing any gamma rays, via the nuclear reaction $Li^6(n,\alpha)t$. The cross section of this neutron absorption reaction is not particularly large at high neutron energies but is very large (i.e., 940 barns) at thermal neutron energies. Married with hydrogen's superior moderating power while still carrying a low mass density, the lithium-6 hydride compound provides the most effective neutron shielding per unit mass or per unit penetration distance of all known materials. Although large power reactor designs typically don't use LiH because cheaper shielding alternatives are available, LiH has been chosen in past reactor designs whenever mass must be minimized. The first excited state of the Li^6 nucleus is 2.185 MeV [Shultix & Faw] so Li^6 may not provide as much inelastic scattering as does Li^7 for fission spectrum neutrons. Therefore, depending on the neutron energy spectrum a lithium hydride with a tailored blend of lithium isotopes may perhaps shield neutrons slightly better than pure lithium-6 hydride.

G.14 Lithium-7 Hydride

This is an isotopically purified version of lithium hydride (LiH). The Li^7 isotope has an extremely low neutron absorption cross section (e.g., 0.033 barns for thermal neutrons). Its nucleus has a first excited state of only 478 keV [Shultix & Faw] so it should provide some inelastic scattering for most fission spectrum neutrons. Married with hydrogen as lithium-7 hydride it could serve as an effective neutron moderator for a thermal neutron spectrum reactor design. The atom densities in lithium-7 hydride are identical to natural lithium hydride, therefore the mass density of lithium-7 hydride is about 101% of natural lithium hydride's mass density which is plotted in Figure G-10.

G.15 Molybdenum

Natural molybdenum occurs in the isotopic abundances: 14.8% Mo-92, 9.3% Mo-94, 15.9% Mo-95, 16.7% Mo-96, 9.6% Mo-97, 24.1% Mo-98, 9.6% Mo-100 [KAPL CoN] Figure G-11 shows radiative capture, elastic scattering, and total cross sections of natural molybdenum. Among the refractory metals compatible at high temperature with molten lithium, molybdenum has the largest experience base in applications with complex shapes. For instance, molybdenum and its alloys are routinely welded. Molybdenum melts at 2896 Kelvins (i.e., 2623 C). Its density is 10.2 g/cm^3. Its linear expansion coefficient is $4.8* 10^{-6}$. Its thermal conductivity is 1.38 W cm^{-1} $Kelvin^{-1}$[CRC].

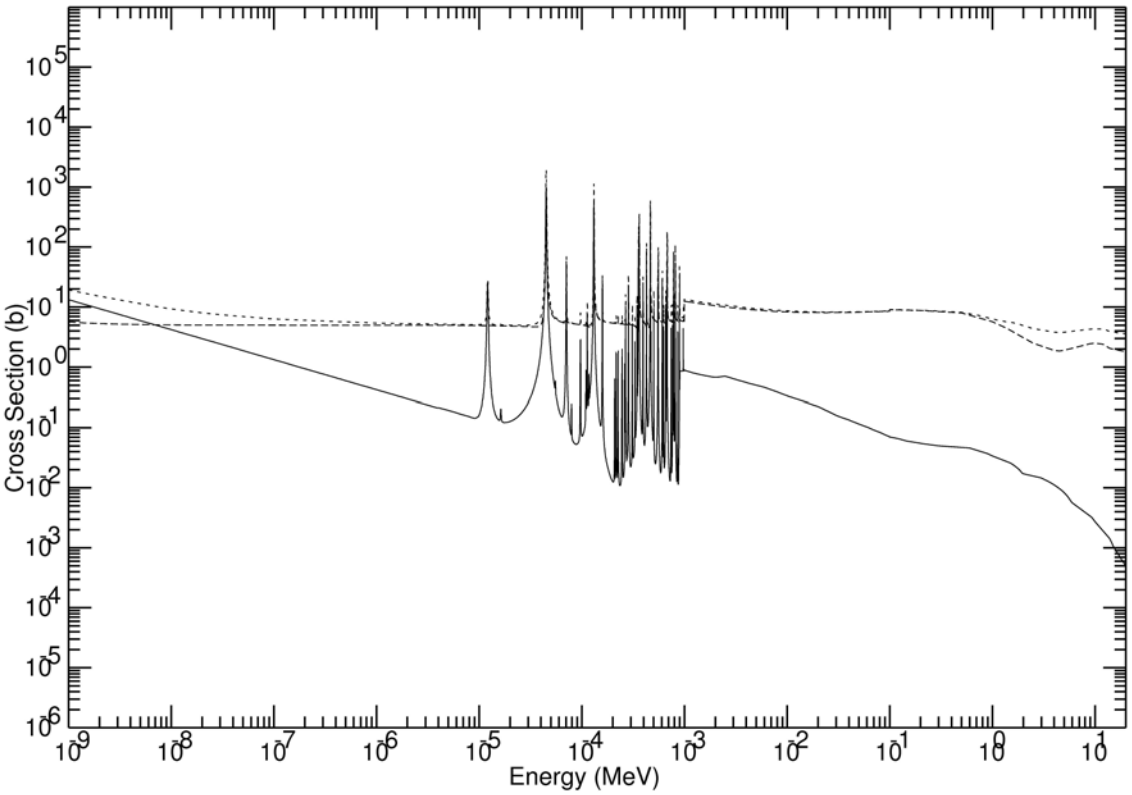


Figure G-11: Capture, Scatter,&Total Cross Sections of Molybdenum
(From http://atom.kaeri.re.kr/ using pointwise ENDF-VI Library)

G.16 Polyethylene

This material is commonly known as plastic. It has the chemical formula $(CH_2)_n$ with large n. It is of interest for nuclear applications simply because it carries a large hydrogen density. With a mass density of 0.92 g/cm^3 it contains hydrogen atoms with a density of $8*10^{22}$ atoms/cm^3. That is 19% more than 1.00 g/cm^3 water with its hydrogen density of $6.7*10^{22}$ atoms/cm^3. As reported in Reactor Shielding, polyethylene softens at 110 C, suffers radiation damage and must be replaced before decomposition due to either temperature or radiation reduces its effectiveness. There are also slightly more dense plastics with reduced hydrogen content which soften at higher temperatures up to 200 C.

G.17 Rhenium

Rhenium is frequently used to form alloys with the lithium-compatible refractory metals. It is a refractory metal in its own right since it melts at 3459 Kelvins (i.e., 3186 C). Its density is 20.8 g/cm^3 and its linear coefficient of expansion is $6.2*10^{-6}$ Kelvin^{-1}. Its thermal conductivity is 0.479 W cm^{-1} Kelvin^{-1}. Naturally occurring rhenium is 37.40% stable Re-185 and 62.60% Re-187 which has an estimated $5*10^{10}$ year half-life. Figure G-12 shows the two isotopes' radiative capture, elastic scattering, and total cross sections.

G.18 Tantalum

Tantalum is compatible with lithium at high temperatures, has good mechanical properties and is resistant to attack from other chemically active agents. Tantalum melts at 3300 Kelvins (3017 C). Its density is 16.4 g/cm^3 and its linear expansion coefficient is $6.3*10^{-6}$/Kelvin. Its thermal conductivity is 0.575 W cm^{-1} Kelvin^{-1}. As tantalum occurs naturally, 99.988% is the completely stable isotope Ta-181. The remaining 0.012% is Ta-180 which has an estimated half-life greater than $1.6*10^{13}$ years. The dominant stable isotope has elastic and total neutron cross sections shown in figure G-13. The deviation between them at low energies represents radiative neutron capture.

G.19Tungsten

Tungsten melts at 3695 Kelvins (i.e., 3422 C), the highest of all metals. It is compatible with lithium. Its density is 19.3 g/cm^3. Its linear coefficient of expansion is $4.5*10^{-6}$ Kelvin^{-1}. Per unit thickness, it attenuates gamma rays better than lead, and its neutron removal cross section exceeds all other materials. The thermal conductivity of tungsten is the highest among the refractory metals, i.e., 1.74 W cm^{-1} Kelvin^{-1} [CRC]. Figure G-14 shows the elastic and total neutron cross sections of natural tungsten. Resonances are resolved below 4 keV and smoothed cross sections are shown above 4 keV.

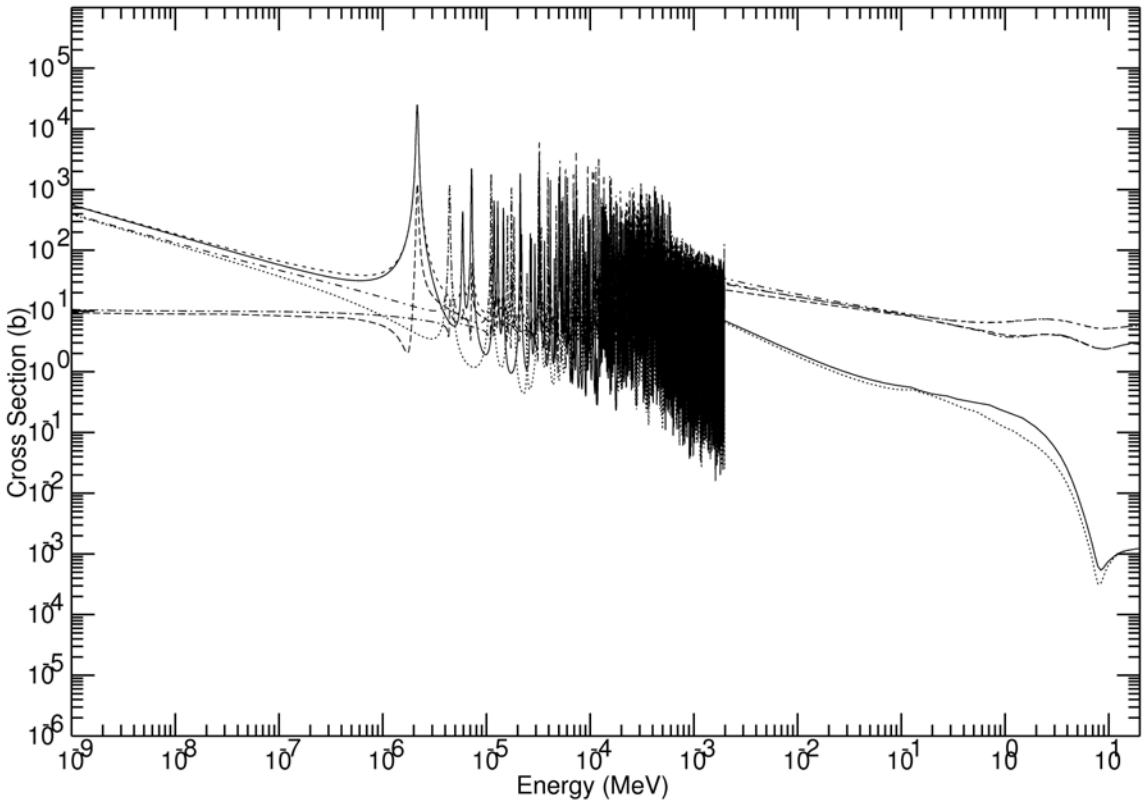


Figure G-12: Rhenium 185 & 187 Capture, Scattering, & Total Cross Sections (From http://atom.kaeri.re.kr/ using ENDF-VI Library)

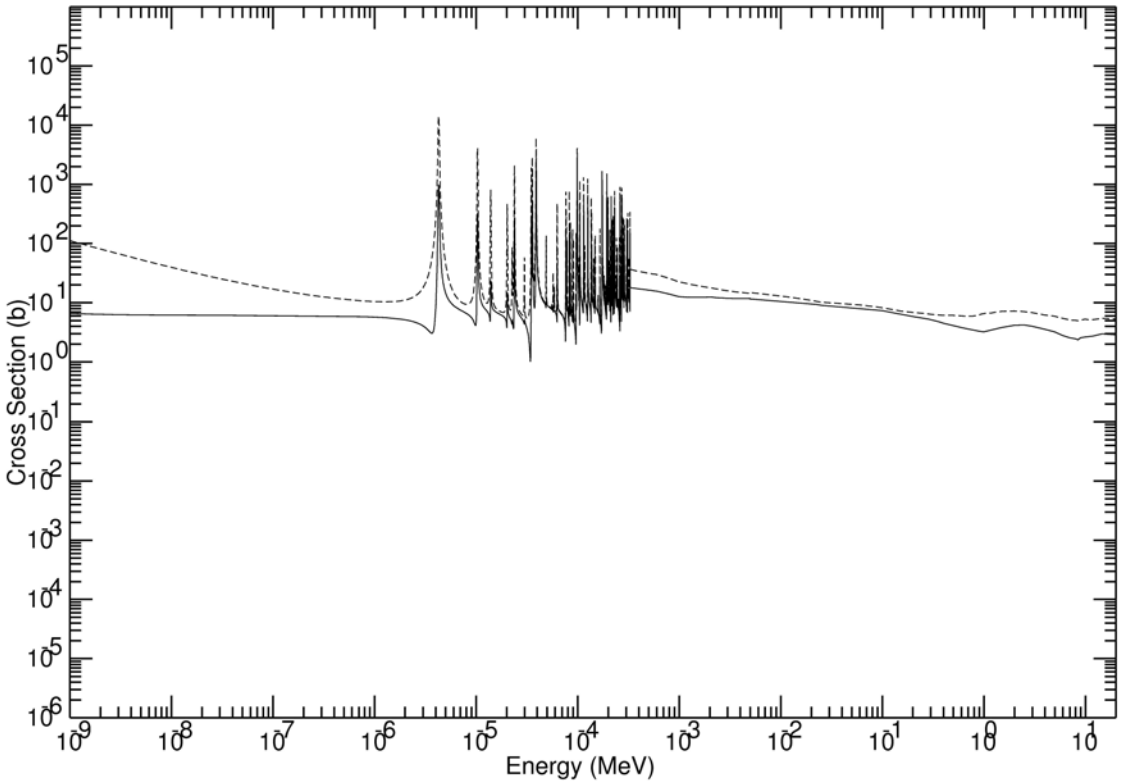


Figure G-13: Scattering and Total Cross Sections Of Ta-181
(From http://atom.kaeri.re.kr/ using ENDF-VI Library)

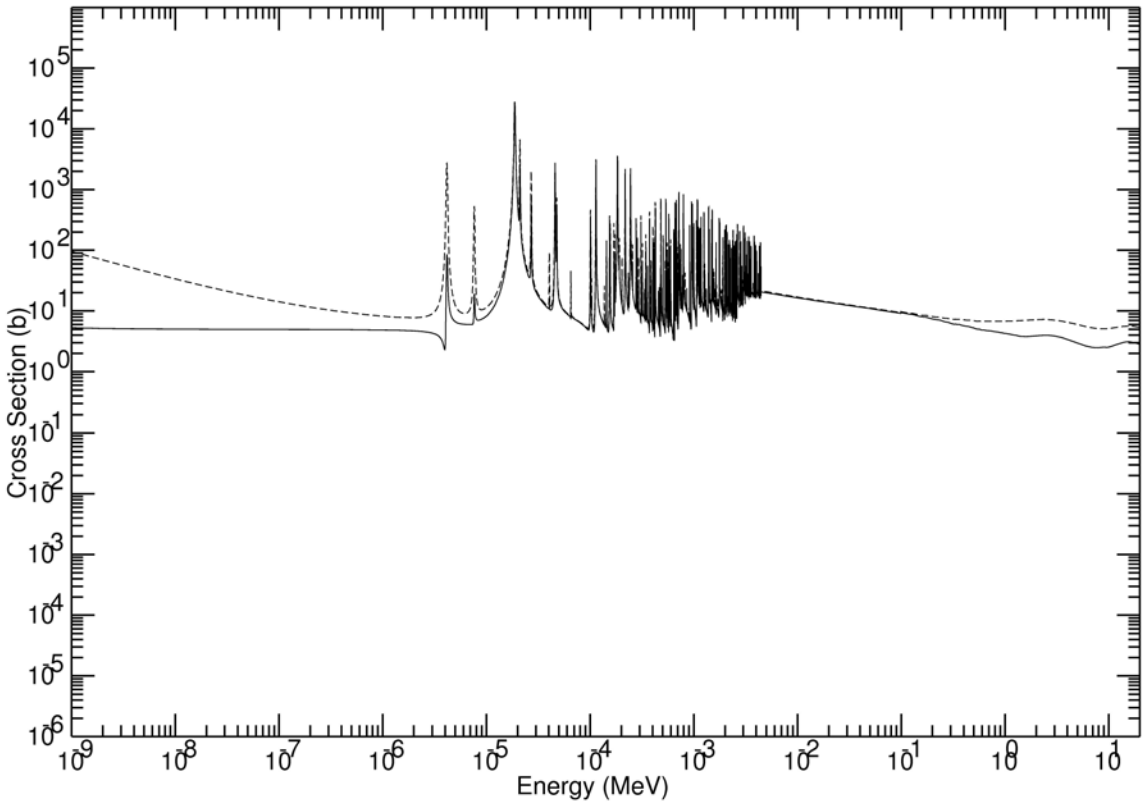


Figure G-14: Scattering and Total Cross Sections Of Tungsten (From http://atom.kaeri.re.kr/ using pointwise ENDF-VI Library)

G.20 Uranium-238

Figure G-15 shows the fission, radiative capture, elastic scattering, and total neutron cross sections of the U-238 isotope. As they appear in the plot, the curves follow that stated sequence from bottom to top. U238 is an excellent gamma ray absorber, surpassing lead.

G.21 Water

Water is abundant on Mars. According to data from orbiting satellites, water is available as near-surface permafrost in many locations in addition to being visible in the polar ice caps. Since water will be used by manned missions to Mars to produce the rocket fuel for their return flights to Earth, it logically follows that water will be available in sufficient quantities so that some of it could be used by the nuclear engines of the present design. Water would most likely be used to optionally augment neutron shielding.

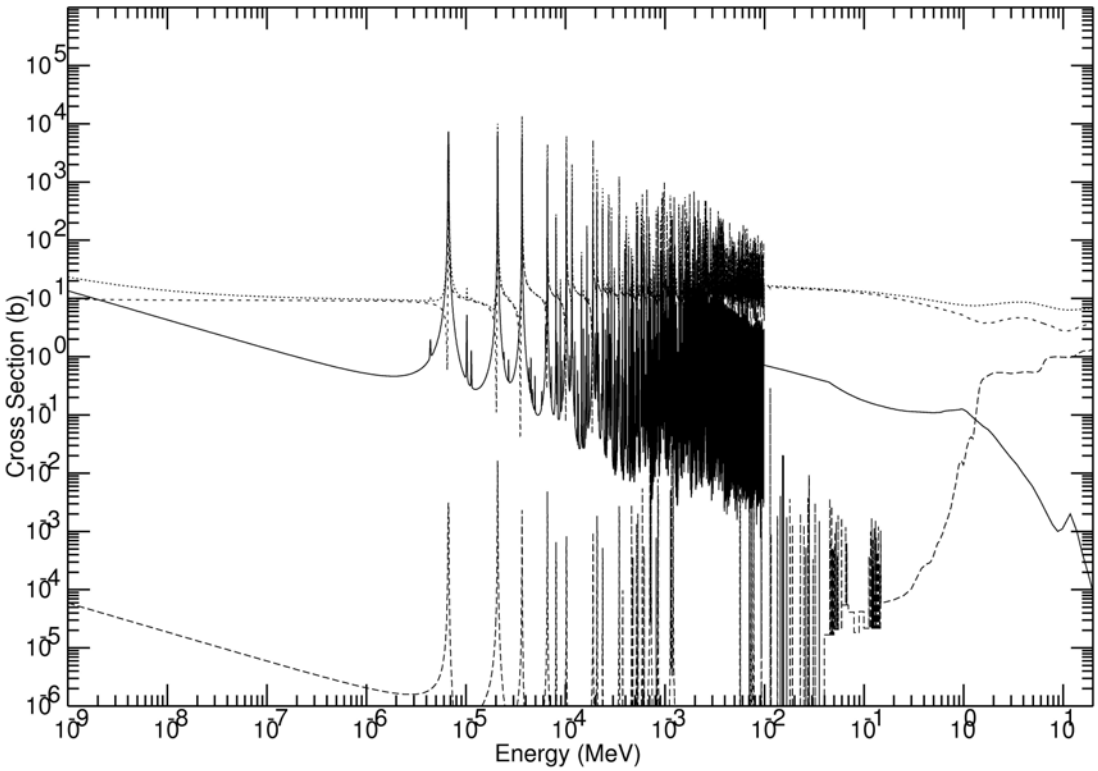


Figure G-15: U-238 Fission, Capture, Scattering, & Total XSections
(From http://atom.kaeri.re.kr/ using ENDF-VI Library)

VITA

Robert D.Woolley was born in 1946. He received a Bachelor of Science degree in Electrical Engineering from Lehigh University in February 1969, a Masters of Science degree in Electrical Engineering from The George Washington University in June 1973, and a Masters of Science degree in Applied Mathematics from Rutgers, the State University of New Jersey, in June 1997. His early engineering career included development of avionics and spacecraft controls while an employee of Fairchild Industries, spacecraft attitude and orbit control services provided to NASA/GSFC while employed by Computer Sciences Corporation, and design of controls for Nuclear Steam Supply Systems while an employee of Combustion Engineering, Inc. Since 1976 he has been employed by the Plasma Physics Laboratory, a national laboratory engaged in thermonuclear fusion research operated by Princeton University for the Department of Energy. He holds three US patents.